Programas de Ciencia e Ingeniería para microcomputadoras SINCLAIR ZX81 COMPATIBLES con el ZX SPECTRUM

CASS LEWART



PROGRAMAS DE CIENCIA E INGENIERIA PARA MICROCOMPUTADORAS SINCLAIR ZX 81 COMPATIBLES CON EL ZX SPECTRUM

PROGRAMAS DE CIENCIA E INGENIERIA PARA MICROCOMPUTADORAS SINCLAIR ZX 81
COMPATIBLES CON EL ZX SPECTRUM

Atlacomulco 499-501, Naucalpan de Juárez, Edo. de México.
Miembro de la Cámara Nacional de la Industria Editorial, Reg. Núm. 465.

ISBN: 968-451-603-7

Traducido de la primera edición en inglés de

SCIENCE ENGINEERING PROGRAMS FOR THE TIMEX/SINCLAIR 1000

ISBN: 0-07-037444-9

Edición exclusiva para ediciones La Colina, S. A. (España)

ISBN: 84-85240-88-X
Depósito legal: M. 9.410-1984

Impreso en Gráficas EMA. Miguel Yuste, 27. MADRID

PRINTED IN SPAIN - IMPRESO EN ESPAÑA

Indice del contenido

PROLOGO

INTRODUCCION

INGENIERIA ELECTRICA

1. Parámetros de circuitos resonantes L-C 1
2. Redes resistivas de atenuación y de adaptación 7
3. Impedancia característica de las líneas de transmisión 12

TRANSMISION DE DATOS

4. Tasas de errores en bits para diversos planes de modulación 18

TEORIA DE LOS NUMEROS

5. Conversión de cualquier base a cualquier otra base 22
6. Conversión hexadecimal a decimal y viceversa 26
7. Descomposición en factores primos 29

PROGRAMACION DE COMPUTADORA

8. Cálculos del desplazamiento (offset) relativo para programación en código máquina 32

TRAZADOS GENERADOS POR COMPUTADORA

9. Alisado de curvas e interpolación/extrapolación con polinomios de Lagrange 36
10. Escalas legibles para trazados por computadora 41

PROBABILIDAD Y ESTADISTICA

11. Combinaciones, permutaciones y factoriales 45
12. La función de error ERF y su función ERFC complementaria ... 49

13. Distribuciones binomia e hipergeométrica 52
14. Desviantes pseudo-aleatorias (gaussianas) normales y uniformes . 57

MATEMATICAS

15. Coeficientes Fourier de funciones periódicas 62
16. Soluciones de ecuaciones diferenciales por el método de Runge-Kutta. 67
17. Soluciones numéricas de ecuaciones transcendentes 71
18. Aritmética de números complejos. 75
19. Paso de las fracciones decimales a fracciones binarias. 80

INVESTIGACION OPERATIVA

20. Solución de ecuaciones de las colas. 83
21. Parámetros de fiabilidad de sistemas complejos 89

MISCELANEA

22. Frecuencia de las notas musicales en la escala bien templada 96
23. Juego de estrategia de inteligencia artificial. 99
24. Tiradas de dados múltiples. 103
25. Cuenta corriente bancaria de dos usuarios. 105

APENDICE

Técnicas y artificios de programación 110
Conversión del Spectrum 113
Referencia y lectura suplementaria. 115
Indice 117

Prólogo

La computadora de bajo coste Timex/Sinclair 1000 suele adquirirse como una primera introducción al campo de las computadoras personales. Pero su aspecto de juguete defrauda a muchos presuntos compradores que no se percatan de su potencial considerable. Prueban unos pocos programas de demostración, encuentran su uso dificultoso y le dejan a un lado. Este libro, escrito específicamente para el Timex/Sinclair 1000, abre nuevos campos de aplicaciones para el estudiante universitario o de segunda enseñanza e incluso para el ingeniero y para el científico.

Un estudiante universitario, o de segunda enseñanza, reconocerá muchos problemas aquí tratados, tales como la aritmética de los números complejos, combinatoria, distribuciones y soluciones a ecuaciones diferenciales y transcendentes, como los temas de sus asignaciones para trabajo en caso. Y los profesionales encontrarán que con estos programas pueden resolverse fácilmente problemas matemáticos y de ingeniería con esta pequeña computadora.

Este libro contiene una colección de programas que personalmente he desarrollado durante varios años para su uso en mi profesión como ingeniero de comunicaciones y para mi entretenimiento en tareas de cálculos caseros y diseño electrónico. Abarcan una amplia gama de problemas en los campos de la ingeniería eléctrica, probabilidad, estadística, teoría de las colas, fiabilidad, ajuste de curvas, generación de gráficos, teoría de los números, informática, inteligencia artificial y otras materias.

Algunos de estos programas se desarrollaron originalmente para las calculadoras programables HP-25 y HP-67 de Hewlett-Packard y se tradujeron después al BASIC. En el proceso de traducción, añadí varias características relativas a una mejor emisión de llamadas orientativas y de contención de errores de entrada. Muchos más programas útiles se escribieron originalmente en BASIC y se añadieron a la colección. Unos pocos programas se han publicado en revistas. En estos casos, se cita la revista de procedencia con los reconocimientos correspondientes.

La codificación se ha desarrollado a partir de las versiones anteriores, menos sofisticadas, concebidas para las computadoras personales Radio Shack/Sharp. La traducción al BASIC del Timex/Sinclair 1000 exigió muchos cambios, pues los lenguajes BASIC de las computadoras Sharp y Timex/Sinclair sólo son parcialmente compatibles. Muchos programas se tuvieron que volver a escribir completamente y hubo de recurrirse al empleo de características funcionales avanzadas del Timex/Sinclair 1000 tales como la manipulación de cadenas y la visualización en pantalla. En el curso de la escritura de los programas del Timex/Sinclair 1000 desarrolló muchos artificios prácticos de programación que se detallan en el apéndice de este libro. Estos artificios, y diversas técnicas de programación descritas

en la sección de Observaciones sobre la Programación de cada capítulo, deben aumentar las aptitudes y conocimientos prácticos de programación del lector.

Todos los programas de este libro están específicamente escritos para la computadora con al menos, 2K de RAM o una computadora ZX-81 análogamente configurada. Sin embargo, cada listado va precedido por una descripción del programa y las ecuaciones fundamentales correspondientes, que deben facilitar la traducción a otras computadoras de BASIC. Cuando se traduzcan a otros lenguajes de computadoras, ha de tenerse cuidado con las «idiosincrasias» de cada lenguaje. Por ejemplo, la función de enteros INT(– 3.5) proporcionará – 4.0 en BASIC, pero dará – 3.0 en Fortran (!).

Espero que encuentre estos programas de utilidad y que le permitirán disfrutar todavía más de su Timex/Sinclair 1000.

Cass Lewart, Holmdel, NJ, 1981

Introducción

Cada capítulo de programa, consta de cinco secciones:

1. Descripción del programa. Proporciona al lector los conocimientos básicos necesarios y la comprensión del problema.

2. Instrucciones. Ponen de manifiesto las secuencias claves requeridas para ejecutar el programa en la computadora Timex/Sinclair 1000.

3. Ejemplos. Sirven para aclarar más el programa. Esta sección incluye párrafos sobre el enunciado del problema, discusión de los resultados y un ejemplo de listado del programa ejecutado directamente por la computadora por medio de la orden COPY.

4. Observaciones sobre la programación. Explican las técnicas de programación y flujo del programa. También sustituyen a las sentencias de comentarios (REM) en los listados que se omiten debido a las limitaciones de memoria de la computadora de 2K.

5. Listados de programas. Se imprimen con la orden LLIST y con interconexión de una impresora de matriz de puntos normal, correspondiendo cada listado exactamente al programa que genera los ejemplos.

Observará que todos los programas incluidos en este libro tienen algunas características comunes. Son fáciles de utilizar y proporcionan unas llamadas, o mensajes, de orientación y una contención de errores de carácter completo. Los programas no son triviales; v.g. no hacen algo que podría hacerse más fácilmente con una calculadora manual. Con muy pocas excepciones, cada programa realiza, sin esfuerzo, funciones que serían difíciles de efectuar de otra forma. Ninguno de los programas es simplemente una traducción de fórmulas. Y no utilizan trucos «indocumentados» para ahorrar un octeto (byte) de memoria a costa de claridad.

Antes de introducir por el teclado los programas pueden tener necesidad de leer la sección del apéndice que describe diversos artificios de programación. Y finalmente, lo último pero no menos importante, ha de leerse el manual del usuario que se suministra con la computadora. Siempre puede encontrar algo de utilidad en su lectura.

INGENIERIA ELECTRICA

1. Parámetros de circuitos resonantes L-C

DESCRIPCION DEL PROGRAMA

Este programa evalúa un circuito resonante que esté constituido por una bobina de capa única, sin núcleo, sin pérdidas y un condensador. El circuito se describe completamente por medio de dos ecuaciones: una fórmula de aproximación para la inductancia de la bobina y una relación de resonancia de un circuito LC.

Fórmula 1-1 $$(2\pi F)^2 \times LC = 1 \qquad L = \frac{(ND)^2 \times 10^{-6}}{18\,D + 40\,A}$$

F es la frecuencia en hertzios, C es la capacidad en faradios, L es la inductancia de la bobina en henrios, A es la longitud de la bobina en pulgadas, D es el diámetro de la bobina en pulgadas y N es el número de espiras de la bobina.

El programa ayuda a diseñar circuitos resonantes, a devanar bobinas y a reparar o modificar receptores de radio y transceptores. Cuando se ejecuta el programa, hay que introducir primero todos los parámetros conocidos y luego dejar que la computadora determine los que faltan por medio de las fórmulas anteriores. Como se muestra en los ejemplos, el programa resolverá problemas bastante complicados. Los parámetros F, C, L, A, D o N pueden introducirse, calcularse o revisarse en orden arbitrario pulsando las teclas correspondientes.

INSTRUCCIONES

El programa genera una petición (mensaje orientativo) de un signo de interrogación (?) para una orden o con el nombre de la función cuando está esperando un valor. Cuando haya aparecido dicho mensaje, ha de pulsarse una de las teclas siguientes:

S	Borrado de todas las variables, comienzo de una nueva ejecución
F	Introducción de la frecuencia F en hertzios
ZF	Cálculo de la frecuencia a partir de C y de L
C	Introducción de la capacidad C en faradios
ZC	Cálculo de la capacidad a partir de F y de L
L	Introducción de la inductancia de la bobina en henrios
ZL	Cálculo de la inductancia a partir de F y de C
ZX o X	Cálculo de la inductancia de la bobina a partir de A, D y N
A	Introducción de la longitud de la bobina en pulgadas
ZA	Cálculo de la longitud de la bobina a partir de L, D y N
D	Introducción del diámetro de la bobina D en pulgadas
ZD	Cálculo del diámetro de la bobina a partir de L, A y N
N	Introducción del número de espiras N para la bobina
ZN	Cálculo de números de espiras a partir de L, A y D

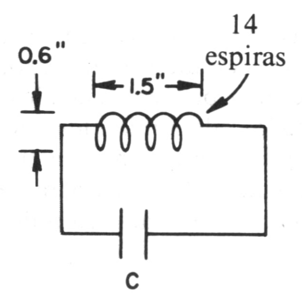
Para revisar el valor de cualquier parámetros, teclear Z seguida por F, C, L, A, D o N.

EJEMPLOS

Enunciados de los problemas

1. Diseñar un circuito resonante que utilice un condensador variable con una capacidad en el margen de 35 pF a 350 pF y una bobina a devanar en un núcleo conformador de 0,6 pulgadas de longitud y 0,3 pulgadas de diámetro. La frecuencia más baja debe ser de unos 6 MHz. Calcular el número de espiras de la bobina y la más alta frecuencia a la que resonará el circuito con el condensador en su valor mínimo.

Figura 1-1



2. Una bobina de una sola capa de 1,5 pulgadas de longitud y 0,6 pulgadas de diámetro, con 14 espiras, como se indica en la figura 1-1, resuena con un condensador desconocido a 10 MHz. ¿Qué valor de condensador en paralelo se requiere para disminuir la frecuencia resonante a 9 MHz?

Ejemplo de Ejecución

```
?F
FREC.?
FREC.=6000000
?C
CAP.?
CAP.=3.5E-10
?Z
CALCULAR
?L
INDUCT.=2.0103409E-6
?A
LONGITUD?
LONGITUD=0.6
?D
DIAM.?
DIAM.=0.3
?Z
CALCULAR
?N
ESPIRAS=25.626381
?N
ESPIRAS?
ESPIRAS=25
?X
INDUCT.=1.9132653E-6
?Z
CALCULAR
?F
FREC.=6150331.3
?C
CAP.?
CAP.=3.5E-11
?Z
CALCULAR
?F
FREC.=19449055
```

```
?F
FREC.?
FREC.=10000000
?A
LONGITUD?
LONGITUD=1.5
?D
DIAM.?
DIAM.=0.6
?N
ESPIRAS?
ESPIRAS=14
?X
INDUCT.=9.9661017E-7
?Z
CALCULAR
?C
CAP.=2.5416453E-10
?

LET T=C

?F
FREC.?
FREC.=9000000
?Z
CALCULAR
?C
CAP.=3.1378337E-10
?

PRINT C-T
5.9618838E-11
```

Discusión de los resultados

El primer problema da como resultado 25,6 espiras. En una aplicación práctica este número se redondea por defecto a 25 espiras. La frecuencia resonante es, entonces, de 6,15 MHz. La más alta frecuencia a la que el circuito resonará con un condensador de 35 pF es, entonces, de 19,44 MHz.

En el segundo problema, calculamos un condensador resonante de 254 pF a 10 MHz. A continuación, interrumpimos la ejecución del programa con BREAK y almacenaremos este valor temporalmente como una variable arbitraria T. La ejecución del programa se reanuda, entonces, con la tecla CONT y se determina la capacidad resonante a 9 MHz (313 pF). El programa vuelve a interrumpirse y la diferencia se encuentra que es de 59 pF, como respuesta final. Este ejemplo ilustra el uso alternativo del Timex/Sinclair 1000 en el modo de programación e inmediato (calculadora).

OBSERVACIONES SOBRE LA PROGRAMACION

El programa ilustra el empleo de la secuencia de INKEY$ con comienzo en la línea 40. La función CODE en la línea 95 proporciona un número (ver página 137-139 del Manual del Usuario) para cada tecla pulsada durante la secuencia de INKEY$. El flujo del programa se dirige, entonces, a un número de línea correspondiente a la tecla pulsada (línea 105). Si la tecla no es reconocida, una rutina de intercepción de errores proporciona un nuevo mensaje orientativo en la línea 30. Un indicador (Z) se pone a «0» o a «1» dependiendo de si el programa espera una entrada (frecuencia, inductancia, etc.) o si debe encontrar un parámetro que falta a partir de los que se dan.

LISTADO DEL PROGRAMA

```
 20 CLS
 22 LET Z=0
 25 SLOW
 27 PAUSE 20
 30 PRINT "?";
 40 IF INKEY$="" THEN GOTO 40
 45 PAUSE 10
 50 LET A$=INKEY$
 60 PAUSE 5
 65 PRINT A$
 70 IF A$=INKEY$ THEN GOTO 70
 95 LET K=CODE A$
100 IF K=56 THEN GOTO 20
```

```
105 IF K=63 OR K=40 OR K=49 OR K=43 THEN
    GOTO K*10
106 IF K=41 OR K=51 OR K=38 OR K=61 THEN
    GOTO K*10
110 GOTO 30
380 IF Z=1 THEN GOTO 384
381 PRINT "LONGITUD?"
382 INPUT A
383 GOTO 386
384 LET Z=0
385 LET A=((D*N*1E-3)**2-18*D*L)/40/L
386 PRINT "LONGITUD=";A
387 GOTO 30
400 IF Z=1 THEN GOTO 404
401 PRINT "CAP.?"
402 INPUT C
403 GOTO 406
404 LET Z=0
405 LET C=1/(2*PI*F)**2/L
406 PRINT "CAP.=";C
407 GOTO 30
410 IF Z=1 THEN GOTO 414
411 PRINT "DIAM.?"
412 INPUT D
413 GOTO 416
414 LET Z=0
415 LET D=(9*L+SQR (81*L*L+40*L*A*N*N/1E-6)
    )/N/N*1E6
416 PRINT "DIAM.=";D
417 GOTO 30
430 IF Z=1 THEN GOTO 434
431 PRINT "FREQ.?"
432 INPUT F
433 GOTO 436
434 LET Z=0
435 LET F=1/2/PI/SQR (L*C)
436 PRINT "FREC.=";F
437 GOTO 30
490 IF Z=1 THEN GOTO 494
491 PRINT "INDUCT.?"
492 INPUT L
493 GOTO 496
494 LET Z=0
495 LET L=1/(2*PI*F)**2/C
496 PRINT "INDUCT.=";L
```

```
 497 GOTO 30
 510 IF Z=1 THEN GOTO 514
 511 PRINT "ESPIRAS?"
 512 INPUT N
 513 GOTO 517
 514 LET Z=0
 515 GOSUB 1000
 516 LET N=SQR (X*L)/D
 517 PRINT "ESPIRAS=";N
 518 GOTO 30
 610 LET Z=0
 611 GOSUB 1000
 612 LET L=(N*D)**2/X
 613 GOTO 496
 630 LET Z=1
 631 PRINT "CALCULAR"
 632 GOTO 30
1000 LET X=(18*D+40*A)*1E6
1010 RETURN
```

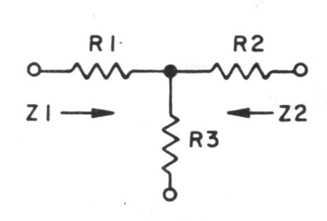
Originalmente publicado con el título de «La calculadora de bolsillo resuelve el problema de la resonancia con el emplo de BASIC». Reimpresión de *Electronics*, 16 junio 1981, Copyright © McGraw-Hill, Inc. 1981. Todos los derechos reservados.

2. Redes resistivas de atenuación y de adaptación

DESCRIPCION DEL PROGRAMA

Este programa calcula los valores de los componentes resistivos para redes de atenuación y de adaptación que suelen utilizarse en redes de transmisión. Calcula las resistencias en serie y en paralelo para seis tipos de redes resistivas: T y «Pi» desequilibradas, O y H equilibradas, y las redes L desequilibrada o C equilibrada adaptadoras de pérdidas mínimas. Además de calcular los valores de las resistencias, el programa determina también las pérdidas mínimas de una red adaptadora de impedancias. Si no se requiere un aislamiento de corriente continua, una red adaptadora de impedancias suele utilizarse como un sustituto sencillo y barato de un transformador adaptador de impedancias. La designación de los componentes se muestra en las figuras, 2-1, 2-2, y 2-3. Si el programa determina valores negativos para las resistencias para una impedancia de entrada, una impedan-

Figura 2-1 Red T desequilibrada/H equilibrada



$$N = 10^{\text{PERDIDAS (dB)}/10}$$

$$R3 = \frac{2\sqrt{N\,Z1 \times Z2}}{N - 1}$$

$$R2 = Z2\left(\frac{N + 1}{N - 1}\right) - R3$$

$$R1 = Z1\left(\frac{N + 1}{N - 1}\right) - R3$$

```
REDES DE ATENUACION
T DESEQ., PI EQUIL. - INTROD.1
PI DESEQ., O EQUIL. - INTROD.2
PERDIDAS MIN. - INTRODUCIR 3?2
INTRODUCIR Z1?300
INTRODUCIR Z2?300
INTRODUCIR PERDIDAS EN DB?10
R1=577.48518 OHM
R2=577.48518 OHM
R3=426.90748 OHM

REDES DE ATENUACION
T DESEQ., PI EQUIL. - INTROD.1
PI DESEQ., O EQUIL. - INTROD.2
PERDIDAS MIN. - INTRODUCIR 3?3
Z1 > Z2
INTRODUCIR Z1?300
INTRODUCIR Z2?75
PERDIDAS=0.57194755 DB
R1=259.80762 OHM
R3=86.60254 OHM

REDES DE ATENUACION
T DESEQ., PI EQUIL. - INTROD.1
PI DESEQ., O EQUIL. - INTROD.2
PERDIDAS MIN. - INTRODUCIR 3?
```

Discusión de los resultados

Por regla general, los valores de resistencias normalizados pueden encontrarse dentro de una pequeña tolerancia porcentual de los valores calculados. Obsérvese que, en el tercer ejemplo, las pérdidas mínimas son de sólo 0,57 dB. Ello podría compararse con el valor entre 1 y 2 dB para un transformador de adaptación.

OBSERVACIONES SOBRE LA PROGRAMACION

Las comprobaciones de errores para los valores de entrada se realizan en la línea 75. A continuación, las fórmulas adecuadas son objeto de selección dependiendo del tipo de red elegido. Para conservar el código se utiliza la misma secuencia de entrada (líneas 90-125) para Z1 y Z2 para los tres tipos de redes. Análogamente, la misma secuencia de salida (impresión) (líneas 350-370) para las tres ramas.

LISTADO DEL PROGRAMA

```
 10 CLS
 30 PRINT "REDES DE ATENUACION"
 40 PRINT "T DESEQ., PI EQUIL. - INTROD.1"
 50 PRINT "PI DESEQ., O EQUIL.,INTROD.2"
 60 PRINT "PERDIDAS MIN. - INTRODUCIR 3?";
 70 INPUT A
 72 PRINT A
 75 IF A<>INT A OR A>3 OR A<1 THEN GOTO 280
 80 IF A=3 THEN PRINT "Z1 > Z2"
 90 PRINT "INTRODUCIR Z1?";
100 INPUT B
105 PRINT B
110 PRINT "INTRODUCIR Z2?";
120 INPUT C
125 PRINT C
130 IF A=3 THEN GOTO 270
140 PRINT "INTRODUCIR PERDIDAS EN DB?";
150 INPUT D
155 PRINT D
160 LET N=10**(D/10)
170 IF A>1 THEN GOTO 220
180 LET G=2*SQR (N*B*C)/(N-1)
190 LET E=B*(N+1)/(N-1)-G
200 LET F=C*(N+1)/(N-1)-G
210 GOTO 350
220 IF A=3 THEN GOTO 270
230 LET G=(N-1)/2*SQR (B*C/N)
240 LET E=1/((N+1)/(N-1)/B-1/G)
250 LET F=1/((N+1)/(N-1)/C-1/G)
260 GOTO 350
270 IF 1-C/B>0 THEN GOTO 300
280 PRINT "ERROR, REINTRODUCIR"
290 GOTO 40
300 LET E=B*SQR (1-C/B)
310 LET G=C/SQR (1-C/B)
320 LET L=LN (SQR (B/C)+SQR (B/C-1))/LN 10
340 PRINT "PERDIDAS=";L;" DB"
350 PRINT "R1=";E;" OHM"
360 IF A<> 3 THEN PRINT "R2=";F;" OHM"
370 PRINT "R3=";G;" OHM"
375 PRINT ""
380 GOTO 20
```

3. Impedancia característica de las líneas de transmisión

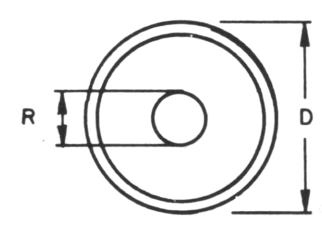
DESCRIPCION DEL PROGRAMA

Este programa calcula la impedancia característica correspondiente a los tipos más comunes de líneas de transmisión: el conductor coaxial simple, la línea blindada equilibrada, la línea bifilar abierta en aire, la línea de bandas paralelas utilizada a las frecuencias de microondas y la línea de transmisión constituida por dos hilos blindados en paralelo con retorno provisto de vaina. Los cinco tipos de líneas de transmisión se muestran en las cinco figuras siguientes con las fórmulas correspondientes. Las unidades de longitud son arbitrarias (pulgadas, pies, centímetros, etc.), pues sólo relaciones de medidas lineales se utilizan en las fórmulas de cálculo.

---

1. Línea coaxial simple

Figura 3-1



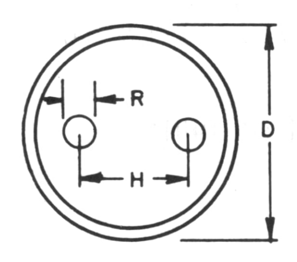
$$Z0 = \frac{138}{\sqrt{\epsilon}} \log \left(\frac{D}{R}\right)$$

ϵ: constante dieléctrica relativa

---

2. Línea blindada equilibrada

Figura 3-2

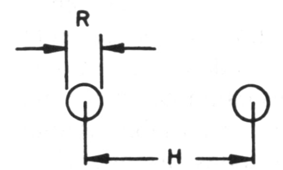


$$Z0 = \frac{276}{\sqrt{\epsilon}} \log \left(2v \frac{1 - \sigma^2}{1 + \sigma^2}\right)$$

$$v = \frac{H}{R} \qquad \sigma = \frac{H}{D}$$

3. Línea bifilar abierta en aire

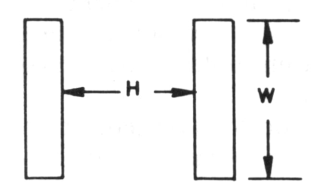
Figura 3-3



$$Z0 = \frac{276}{\sqrt{\epsilon}} \log\left(\frac{H}{R} + \sqrt{\left(\frac{H}{R}\right)^2 - 1}\right)$$

4. Línea de bandas paralelas

Figura 3-4



$$Z0 = \frac{377}{\sqrt{\epsilon}} \times \frac{H}{W}$$

$$H/W < 0.1$$

5. Dos hilos en paralelo con retorno provisto de vaina

Figura 3-5 La misma figura que figura 3-2

$$Z0 = \frac{69}{\sqrt{\epsilon}} \log\left(\frac{v}{2\sigma^2}(1 - \sigma^4)\right)$$

$\sigma = H/D$ $v = H/R$

LISTADO DEL PROGRAMA

```
 15 CLS
 20 PRINT "IMPEDANCIA CARACT."
 25 PRINT
 27 PRINT "INTRODUCIR TIPO (1-5)?";
 30 INPUT A
 35 PRINT A
 37 IF A<>INT A OR A<1 OR A>5 THEN GOTO 60
 40 PRINT "INTROD. CONST. DIEL. REL.?";
 45 INPUT E
 50 PRINT E
 55 IF E>0 THEN GOTO 100*A
 60 LET L=25
 65 GOTO 720
100 PRINT "LINEA COAXIAL SIMPLE"
110 GOSUB 650
120 IF R>0 AND D>R THEN GOTO 150
130 LET L=110
140 GOTO 720
150 LET Z=138/SQR E*LN (D/R)/LN 10
160 GOTO 580
200 PRINT "LINEA BLINDADA EQUILIBRADA"
210 GOSUB 600
220 IF R>0 AND H>R AND D>H+R THEN GOTO 250
230 LET L=210
240 GOTO 720
250 LET V=H/R
260 LET S=H/D
270 LET Z=276/SQR E*LN (2*V*(1-S*S)/
    (1+S*S))/LN 10
280 GOTO 580
300 PRINT "LINEA BIFILAR"
310 GOSUB 600
320 IF R>0 AND H>R THEN GOTO 350
330 LET L=310
340 GOTO 720
350 LET S=H/R
360 LET Z=276*LN (S+SQR (S*S-1))/LN 10/SQR E
370 GOTO 580
400 PRINT "LINEA BANDAS PARALELAS"
410 PRINT "ANCHURA CONDUCTOR W?";
420 INPUT W
430 PRINT W
440 GOSUB 600
```

```
450 IF>H 0 AND W>0 THEN GOTO 480
460 LET L=410
470 GOTO 720
480 LET Z=377*H/W/SQR E
490 GOTO 580
500 PRINT "2 LINEAS PAR., RET. CON VAINA"
510 GOSUB 600
520 IF R>0 AND H>R AND D>H+R THEN GOTO 550
530 LET L=510
540 GOTO 720
550 LET V=H/R
560 LET S=H/D
570 LET Z=69/SQR E*LN (V/2/S/S*(1-S**4))/LN 10
580 PRINT "ZO=";Z;" OHM"
590 GOTO 25
600 PRINT "SEPARACION H?";
610 INPUT H
620 PRINT H
630 IF A=3 THEN GOTO 680
640 IF A=4 THEN RETURN
650 PRINT "DIAM. EXT. D?";
660 INPUT D
670 PRINT D
680 PRINT "DIAM. COND. R?";
690 INPUT R
700 PRINT R
710 RETURN
720 PRINT "ERROR, REINTRODUCIR"
730 GOTO L
```

TRANSMISION DE DATOS

4. Tasas de errores en bits para diversos planes de modulación

DESCRIPCION DEL PROGRAMA

En un enlace de transmisión de datos bien diseñado, el ruido aleatorio con distribución gausiana es la fuente principal de errores de bits de datos. La probabilidad de recibir un bit de datos incorrecto depende de la relación señal/ruido (S/N) y del tipo de modulación de datos utilizada. Por regla general, la complejidad del circuito está ligada con la probabilidad de error. Un circuito más complejo, con más estrictas exigencias de atenuación y de distorsión de retardo dará lugar a una menor probabilidad de error para la misma relación S/N. El siguiente programa calcula la probabilidad de error de bit (Pe) como una función de S/N para nueve planes de modulación bien cómodos. Las fórmulas utilizadas son las siguientes:

---

1. Manipulación todo-nada (On-Off Keying = OOK), coherente:

Fórmula 4-1

$$Pe = \frac{1}{2}\,\mathrm{Erfc}\left(\frac{1}{2}\sqrt{\frac{S}{N}}\right)$$

---

2. Manipulación por desplazamiento de frecuencia (FSK), coherente
   Manipulación por desplazamiento de amplitud (ASK), coherente
   Modulación por codificación de impulsos (PCM), unipolar

Fórmula 4-2

$$Pe = \frac{1}{2}\,\mathrm{Erfc}\sqrt{\frac{1}{2}\,\frac{S}{N}}$$

---

3. PCM, polar
   Manipulación por desplazamiento de fase (PSK)

Fórmula 4-3

$$Pe = \frac{1}{2} \text{Erfc} \sqrt{\frac{S}{N}}$$

4. FSK, no coherente

Fórmula 4-4

$$Pe = \frac{1}{2} e^{\left(-\frac{S}{2N}\right)}$$

5. Manipulación por desplazamiento de fase diferencial (DPSK)

Fórmula 4-5

$$Pe = \frac{1}{2} e^{\left(-\frac{S}{N}\right)}$$

6. ASK, no coherente

Fórmula 4-6

$$Pe = \frac{1}{2} \text{Erfc} \left(\frac{1}{2} \sqrt{\frac{S}{N}}\right)$$

La función Erfc (función de error complementaria) relacionada con la distribución de ruido gaussiano se define como sigue. La función se calcula de forma recurrente, como se indica en el programa 12.

Fórmula 4-7

$$\text{Erf}(x) = \frac{2}{\sqrt{\pi}} \int_0^x e^{(-t^2)}\, dt \qquad \text{Erfc}(x) = 1 - \text{Erf}(x)$$

INSTRUCCIONES

Cuando se pulsa la tecla RUN, el programa mide la relación señal/ruido en dB y la clase de modulación. Seleccionar una relación S/N y un número entre 1 y 6 para indicar la clase de modulación tal como se describe en la descripción del programa. Entonces, el programa calcula la probabilidad de error de bits.

EJEMPLOS

Enunciado de problema

1. Calcular la probabilidad de error de bit Pe para una S/N de 12 dB para nueve planes de modulación.

Ejemplo de ejecución

```
PROBABILIDAD DE ERROR DE BIT
INTROD. RELACION S/N EN DB?12
SELECCIONAR CODIGO MODUL.1-6?1
PROB(12 DB)=.0024385373

INTROD. RELACION S/N EN DB?12
SELECCIONAR CODIGO MODUL.1-6?2
PROB(12 DB)=.000034302624

INTROD. RELACION S/N EN DB?12
SELECCIONAR CODIGO MODUL.1-6?3
PROB(12 DB)=9.0060101E-9

INTROD. RELACION S/N EN DB?12
SELECCIONAR CODIGO MODUL.1-6?4
PROB(12 DB)=.0001808915

INTROD. RELACION S/N EN DB?12
SELECCIONAR CODIGO MODUL.1-6?5
PROB(12 DB)=6.544347E-8

INTROD. RELACION S/N EN DB?12
SELECCIONAR CODIGO MODUL.1-6?6
PROB(12 DB)=.0001990186
```

Discusión de resultados

La probabilidad de error para los nueve planes de modulación varía mucho. La más baja probabilidad de error se da por el plan de modulación tipo 3 (PCM, Polar/PSK) y la más alta por el plan de modulación tipo 1 (OOK, coherente).

OBSERVACIONES SOBRE LA PROGRAMACION

Las fórmulas aparentemente diferentes se resuelven por la misma sección de código recalculando el argumento de la función de error complementaria. Una nueva discusión de evaluación recurrente de la función de error se da en el programa 12.

LISTADO DEL PROGRAMA

```
15 CLS
20 PRINT "PROBABILIDAD DE ERROR BIT"
```

```
 25 PRINT "INTROD. RELACION S/N EN DB?";
 30 INPUT B
 35 PRINT B
 40 LET C=B
 45 PRINT "SELECCIONAR CODIGO MODUL.1-6?";
 50 INPUT D
 55 PRINT D
 60 IF D>0 AND D<7 AND D=INT D THEN GOTO 100*D
 65 PRINT "ERROR ENTRADA"
 70 GOTO 45
100 GOSUB 800
200 GOSUB 800
300 LET X=10**(C/20)
305 LET Y=1/SQR PI*EXP (-X*X)
310 IF X<2 THEN GOTO 350
315 LET A=14/X
320 FOR I=27 TO 1 STEP -1
325 LET A=I/2/(X+A)
330 NEXT I
335 LET Z=Y/(X+A)
340 LET N=1-Z
345 GOTO 700
350 LET I=1
355 LET T=2*X*Y
360 LET N=T
365 LET T=2*X*X*T/(2*I+1)
370 LET M=N+T
375 IF M=N THEN GOTO 393
380 LET N=M
385 LET I=I+1
390 GOTO 365
393 LET Z=1-N
396 GOTO 700
400 GOSUB 800
500 LET Z=EXP -(10**(C/10))
510 GOTO 700
600 LET F=10**(C/10)
610 LET Z=EXP (-F/2)*(1+1/SQR (2*PI*F))
700 LET Z=Z/2
710 PRINT "PROB(";B;" DB)=";Z
720 PRINT
730 GOTO 25
800 LET C=C-10*LN 2/LN 10
810 RETURN
```

TEORIA DE LOS NUMEROS

5. Conversión de cualquier base a cualquier otra base

DESCRIPCION DEL PROGRAMA

La conversión de números entre diferentes bases se suele realizar por programadores de computadoras, matemáticos, estudiantes de segunda enseñanza y universitarios. En particular, el desarrollo de un programa de computadora exige la conversión entre las bases de numeración binaria (2), octal (8), decimal (10) y hexadecimal (16). Los problemas de la teoría de números pueden requerir la conversión entre éstos y otras bases.

La conversión entre dos bases arbitrarias a y b suele ser un procedimiento de dos pasos, que convierte primero desde la base *a* a la base 10 y luego, desde la base 10 a la base *b*. El método singular aquí presentado para enteros positivos utiliza el mismo algoritmo para ambas conversiones, con el ahorro consiguiente en pasos de programación.

Para representar un número en una base más grande que 10, se necesita más de una posición para representar un dígito en esa base. Por ejemplo, 17 FE (base 16) se introduciría o visualizaría como [01], [07], [15], [14] o simplemente 1071514 (1 = 01, 2 = 02,..., A = 10, B = 11, C = 12, D = 13, E = 14, F = 15). Por lo general, una sola posición se reservaría por dígito para las bases 2-10, dos posiciones para bases 11-100, etc. El programa interpreta, de forma automática, la entrada o la visualización dependiendo de la base. El programa no tiene ninguna limitación inherente sobre la magnitud de la base o el número a convertir. El único factor limitador es la precisión de la computadora. Por ejemplo, la precisión del Timex/Sinclair BASIC limitará la entrada y la salida a unas 10 posiciones decimales. Un número más grande daría lugar a un error.

INSTRUCCIONES

Después de pulsar la tecla RUN, hay que atender las peticiones del programa introduciendo el número a convertir (argumento) y las bases antigua

y nueva. El programa visualizará el argumento convertido a la nueva base; si ni la base antigua, ni la nueva, es igual a 10, entonces, se visualizará también el argumento en la base 10.

EJEMPLOS

1. Convertir 17C (base 16) a base 2.
2. Convertir el equivalente decimal de 17C a base 2 (ambos resultados deben dar la misma respuesta).
3. Convertir el argumento decimal 12345 a base 122.
4. Convertirlo de nuevo desde base 122 a base 10.

Ejemplo de ejecución

```
CONVERSION BASE GENERAL
INTROD. ARGUMENTO?10712
INTROD. BASE ANTIGUA?16
INTROD. BASE NUEVA?2
10712 A BASE 16
=1011111 A BASE 2
=380 A BASE 10

INTROD. ARGUMENTO?380
INTROD. BASE ANTIGUA?10
INTROD. BASE NUEVA?2
380 A BASE 10
=101111100 A BASE 2

INTROD. ARGUMENTO?12345
INTROD. BASE ANTIGUA?10
INTROD. BASE NUEVA?122
12345 A BASE 10
=101023 A BASE 122

INTROD. ARGUMENTO?101023
INTROD. BASE ANTIGUA?122
INTROD. BASE NUEVA?16
101023 A BASE 122
=3000309 A BASE 16
=12345 A BASE 10
```

Discusión de los resultados

Se requiere alguna interpretación de los resultados para bases mayores que 10. El resultado del tercer ejemplo debe interpretarse como [101], [023] (base 122).

OBSERVACIONES SOBRE LA PROGRAMACION

Una amplia verificación de errores de los datos de entrada se realiza en las líneas, 70, 110 y 150. El error de redondeo se hace mínimo utilizando la función INT y sumando 0,5 a las funciones adecuadas en las líneas 440 y 480. La línea 480 determina el número de dígitos decimales requeridos para representar cada «dígito» para una base arbitraria.

LISTADO DEL PROGRAMA

```
 20 CLS
 30 PRINT "CONVERSION BASE GENERAL"
 40 PRINT "INTROD. ARGUMENTO?";
 50 INPUT C
 60 PRINT C
 70 IF C<>INT (ABS C) THEN GOTO 40
 80 PRINT "INTROD. BASE ANTIGUA?";
 90 INPUT D
100 PRINT D
110 IF D<>INT (ABS D) THEN GOTO 80
120 PRINT "INTROD. BASE NUEVA?";
130 INPUT E
140 PRINT E
150 IF E<>INT (ABS E) THEN GOTO 120
160 IF D<>10 THEN GOTO 190
170 LET N=C
180 GOTO 250
190 LET U=D
200 GOSUB 480
210 LET S=C
220 LET Q=D
230 LET R=V
240 GOSUB 390
250 IF E<>10 THEN GOTO 280
260 LET S=N
270 GOTO 340
280 LET U=E
290 GOSUB 480
```

```
300 LET S=N
310 LET Q=V
320 LET R=E
330 GOSUB 390
340 PRINT C;" A BASE ";D
350 PRINT "=";N;" A BASE ";E
360 IF D<>10 AND E<>10 THEN PRINT "=";S;"
    TO BASE 10"
370 PRINT " "
380 GOTO 40
390 LET M=0
400 LET N=0
410 LET P=S
420 LET T=P
430 LET P=INT (P/R)
440 LET N=N+INT ((T-P*R)*Q**M+.5)
450 IF P=0 THEN RETURN
460 LET M=M+1
470 GOTO 420
480 LET V=INT (10**(1+INT (LN (U-1)/LN 10))+.5)
490 RETURN
```

6. Conversión hexadecimal a decimal y viceversa

DESCRIPCION DEL PROGRAMA

Este programa difiere del anterior programa de conversión de cualquier base en que sólo convierte entre bases 10 y 16. Las dos conversiones (10 a 16 y 16 a 10) son muy importantes para desarrollo de programa de computadora, pues la mayor parte de los dispositivos de programación (ensambladores, mapas de memoria, desensambladores, etc.) se refieren a las posiciones de memoria y al contenido de memoria en base 10 ó 16 (hex). Este programa, a diferencia con el anterior, utiliza la presentación alfanumérica estándar para dígitos hexadecimales: 0 a 9 inclusive y A, B, C, D, E y F, para los dígitos 10 a 15 inclusive. Sigue siendo aplicable el comentario acerca de la precisión y del redondeo del programa anterior.

INSTRUCCIONES

Pulsar la tecla RUN y seguir las indicaciones (mensajes orientativos) para realizar la conversión hexadecimal-decimal (1) o decimal-hexadecimal (2). Después de la siguiente indicación, introducir el argumento en hexadecimal (empleando A a F, si fuera necesario) o como un número decimal.

EJEMPLOS

1. Convertir F697 (base 16) a base 10.
2. Convertir el resultado de nuevo a base 16.
3. Convertir FFFF (base 16) a base 10.

Ejemplo de ejecución

```
HEX A DECIMAL, INTROD.1
DECIMAL A HEX, INTROD.2?1
INTROD. ARGUMENTO EN HEX?F697
=63127 EN BASE 10

HEX A DECIMAL, INTROD.1
DECIMAL A HEX, INTROD.2?2
```

```
INTROD. ARGUMENTO A BASE 10?63127
=F697 EN BASE 16

HEX A DECIMAL, INTROD.1
DECIMAL A HEX, INTROD.2?1
INTROD. ARGUMENTO EN HEX?FFFF
=65535 EN BASE 10
```

Discusión de los resultados

Las indicaciones y los resultados son auto-explicatorios.

OBSERVACIONES SOBRE LA PROGRAMACION

La comprobación de errores de los datos de entrada se realiza en las líneas 70 y 270. Los dígitos en base 16 se introducen como una cadena de caracteres (A$) y cada carácter en la cadena se interpreta en la línea 260. Si se encuentra un carácter que no corresponde a un número entre 0 y 15, se rechaza como un error. La traducción desde números decimales a una cadena de caracteres en hexadecimal se produce en la línea 510. Este programa pone de manifiesto la potencia de las funciones de manipulación de cadenas de caracteres en BASIC, tales como CHR$, CODE y LEN.

LISTADO DEL PROGRAMA

```
 10 DIM B$(10)
 20 CLS
 30 PRINT
 35 PRINT "HEX A DECIMAL, INTROD.1"
 40 PRINT "DECIMAL A HEX, INTROD.2?";
 50 INPUT B
 60 PRINT B
 70 IF B<>1 AND B<>2 THEN GOTO 30
 80 GOTO 200*B
200 PRINT "INTROD. ARGUMENTO EN HEX?";
210 INPUT A$
220 PRINT A$
230 LET L=LEN A$
240 LET N=0
250 FOR I=0 TO L-1
260 LET C=CODE A$(L-I)-28
270 IF C>=0 AND C<=16 THEN GOTO 310
280 PRINT "ENTRADA ERRONEA"
```

```
290 PAUSE 200
300 RUN
310 LET N=N+C*16**I
320 NEXT I
330 PRINT "=";N;" EN BASE 10"
340 GOTO 640
400 PRINT "INTROD. ARGUMENTO A BASE 10?";
410 INPUT D
420 PRINT D
430 IF D=INT (ABS D) THEN GOTO 470
440 PRINT "ENTRADA ERRONEA"
450 PAUSE 200
460 RUN
470 LET K=0
480 LET K=K+1
485 LET F=D
490 LET D=INT (D/16)
500 LET E=F-16*D
510 LET B$(K)=CHR$ (E+28)
580 IF F>0 THEN GOTO 480
590 PRINT "=";
600 FOR I=1 TO K-1
610 PRINT B$(K-I);
620 NEXT I
630 PRINT "EN BASE 16"
640 PRINT " "
650 GOTO 30
```

7. Descomposición en factores primos

DESCRIPCION DEL PROGRAMA

Uno de los primeros problemas con los que se enfrenta un estudiante en la teoría de los números es la descomposición en primos de un entero. Esta operación no es simplemente un ejercicio teórico (es aplicable en muchas áreas incluyendo la criptografía). Algunos de los modernos códigos criptográficos están basados en el supuesto de que lleva mucho tiempo, incluso para computadoras grandes, determinar los factores primos de números grandes. En la técnica denominada «clave pública», puede publicarse el código de cifrado (producto de dos primos), mientras que los factores primos que constituyen el código de descifrado son conocidos solamente para los receptores autorizados. Si un código está basado en una clave que es un producto de dos primos grandes, lo que llevaría años a una computadora determinar, el código será inviable a todos los efectos prácticos.

Los dos métodos principales para encontrar los factores primos de un entero son la criba de Eratóstenes, así llamada en honor a un famoso matemático griego y una sencilla comprobación de divisores impares. El primer método es más rápido pero también más adecuado para grandes computadoras, pues almacena en memoria todos los primos sucesivos más pequeños que la raíz cuadrada del número a descomponer. El segundo método es mucho más sencillo, pues solamente comprueba la descomposición en 2, 3, 5 y los sucesivos números enteros impares. Aunque el segundo método desperdicia algo de tiempo de ejecución, puesto que comprueba todos los números impares y no sólo los números enteros primos, tiene menos necesidades de memoria en la computadora, porque no han de almacenarse resultados intermedios. Por consiguiente, el segundo método es el único adecuado para el cálculo de números primos en la computadora Timex/Sinclair 1000 y es la base para el programa siguiente.

INSTRUCCIONES

Pulsar la tecla RUN para iniciar el programa y luego, seguir las indicaciones introduciendo el número a descomponer en factores primos. A continuación, el programa visualiza cuantos números primos se encontraron y dichos factores primos, uno por uno.

EJEMPLOS

Enunciado del problema

Descomponer 12345, 66759 y 71 en factores primos.

Ejemplo de ejecución

```
BUSCADOR FACTORES PRIMOS
INTRODUCIR NUMERO?12345
 NO. DE FACTORES=3
FACTOR NO. 1=3
FACTOR NO. 2=5
FACTOR NO. 3=823

BUSCADOR FACTORES PRIMOS
INTRODUCIR NUMERO?66759
 NO. DE FACTORES=5
FACTOR NO. 1=3
FACTOR NO. 2=7
FACTOR NO. 3=11
FACTOR NO. 4=17
FACTOR NO. 5=17

BUSCADOR FACTORES PRIMOS
INTRODUCIR NUMERO?71
 NO. DE FACTORES=1
FACTOR NO. 1=71
```

Discusión de resultados

Obsérvese, en el segundo ejemplo, que 17 aparece dos veces, lo que indica que es un factor múltiple. En el tercer ejemplo, probamos un número primo (71) que tiene un factor no trivial único, que es el mismo.

OBSERVACIONES SOBRE PROGRAMACION

El programa realiza la comprobación de errores en la línea 90. Si el número a descomponer es más grande que 1.000.000.000, los errores del redondeo no permitirán que el programa actúe correctamente. Para llamada del usuario los cincuenta primeros factores primos están almacenados en una matriz («array») dimensionada B(50) y podrían llamarse por medio de un programa escrito por el usuario. Si hay más de 50 factores, el programa visualizará solamente los cincuenta primeros.

LISTADO DEL PROGRAMA

```
 20 CLS
 25 LET K=0
 27 LET D=0
 30 DIM B (50)
 40 PRINT "BUSCADOR FACTORES PRIMOS"
 50 PRINT "INTRODUCIR NUMERO?";
 60 INPUT A
 70 PRINT A
 80 LET F=A
 90 IF A=INT A AND A>1 AND A<1E9 THEN.GOTO 120
100 PRINT "ERROR, REINTRODUCIR"
110 GOTO 50
120 LET E=A/2
130 LET C=2
140 IF E<>INT E OR 2*E<A THEN GOTO 170
150 GOSUB 320
160 GOTO 120
170 LET D=D+1
180 LET C=1+2*D
190 IF C>SQR A THEN GOTO 250
200 LET E=A/C
210 IF E<>INT E OR E*C<>A THEN GOTO 170
220 LET D=D-1
230 GOSUB 320
240 GOTO 170
250 IF A<=1 THEN GOTO 280
260 LET K=K+1
270 LET B(K)=A
280 PRINT " NO. DE FACTORES=";K
290 FOR I=1 TO K
300 PRINT "FACTOR NO. ";I;"=";B(I)
310 NEXT I
312 PRINT " "
315 GOTO 25
320 IF K>=50 THEN RETURN
330 LET K=K+1
340 LET B(K)=C
350 LET A=E
360 RETURN
```

PROGRAMACION DE COMPUTADORA

8. Cálculos del desplazamiento (offset) relativo para programación en código máquina

DESCRIPCION DEL PROGRAMA

La finalidad de este programa es calcular las direcciones relativas de instrucciones en código máquina. Cuando se escriben programas cortos en código máquina para cualquiera de los microprocesadores de 8 bits más conocidos, tales como Z80, 6502 o 6800, uno de ellos ha de cargar el programa ensamblador, que puede no estar disponible, o ensamblar «a mano» el programa en código máquina. El problema principal es, entonces, encontrar los argumentos de las instrucciones de bifurcación relativa. El programa aquí mostrado determinará, en notación hexadecimal, el argumento de una instrucción de bifurcación relativa ZZ dada la dirección absoluta de esa instrucción, la dirección «FROM» (X) y la dirección hexadecimal de la instrucción a la que el programa debe bifurcar si es verdadera la condición de bifurcación, la dirección «TO» (Y). Por ejemplo, dado el siguiente código en lenguaje máquina del Z80:

Dirección	Instrucción	Rótulo	Nemónico
20FC	DD 7E 00	LOOP	LD A,(IX)
....	....	....	....
2109	B8		CP B
210A	20 zz		JR NZ LOOP

se ha de encontrar el valor de zz, también conocido como desplazamiento (offset), dada la dirección de la instrucción de bifurcación «FROM» 210A (hex) y la dirección de la instrucción «TO»20FC (hex). Para calcular zz, debe resolverse una de las ecuaciones siguientes:

zz = Y – X – 2 para Y > X (bifurcación hacia adelante) o
zz = FF (hex) + Y – X – 1 para Y < = X (bifurcación hacia atrás)

El valor offset resultante en el anterior ejemplo sería FO (hex) y la última instrucción leería 20F0. Habida cuenta de que el offset zz sólo puede tomar valores entre 0 y FF (hex) para microprocesadores de 8 bits, la bifurcación relativa está limitada a media página (1 página = 256 posiciones) en el sentido hacia adelante o hacia atrás. El programa comprueba si se puede realizar la instrucción de bifurcación, pues analiza las direcciones de las instrucciones FROM/TO. El programa indicará si la bifurcación relativa puede, o no puede, realizarse en un solo paso y si dará lugar a un salto hacia adelante o hacia atrás.

INSTRUCCIONES

Cuando se pulsa la tecla RUN, el programa realiza la llamada de la dirección de las instrucciones «FROM» y «TO». Ambas deben introducirse en notación hexadecimal.

EJEMPLOS

Enunciado del problema

Ejemplo n.º	Instrucción bifurcación dirección absoluta (hex)	Instrucción de destino dirección absoluta (hex)
1	123A	127B
2	25F3	2613
3	010A	00FC
4	AA25	BA24

Ejemplo de ejecución

```
DIRECCIONAMIENTO REL. (8 BITS)

INTROD. DIREC. "FROM" EN HEX?123A
INTROD. DIREC. "TO" EN HEX?127B
OFFSET ADELANTE=3F
```

```
INTROD. DIREC. "FROM" EN HEX?25F3
INTROD. DIREC. "TO" EN HEX?2613
OFFSET ADELANTE=1E

INTROD. DIREC. "FROM" EN HEX?10A
INTROD. DIREC. "TO" EN HEX?FC
OFFSET ATRAS O EN LUGAR=F0

INTROD. DIREC. "FROM" EN HEX?AA25
INTROD. DIREC. "TO" EN HEX?BA24
4095(DEC) OFFSET DEM. GRANDE,REINT.

INTROD. DIREC. "FROM" EN HEX?AA25
INTROD. DIREC. "TO" EN HEX?AA23
OFFSET ATRAS O EN LUGAR=FC
```

Discusión de resultados

Los tres primeros ejemplos dan lugar a offsets hexadecimales de 3F, 1E y FO. El cuarto ejemplo indica que el offset es demasiado grande. Nos percatamos, entonces, de que la dirección «TO» debe ser AA23 en lugar de BA24 y finalizar con un offset de FC.

OBSERVACIONES SOBRE PROGRAMACION

Obsérvese el empleo de dobles comillas (Q desplazada) en las líneas 45 y 120. Para realizar cálculos aritméticos en número hexadecimales, los números se convierten primero a la base 10 y los resultados se vuelven a convertir a base 16. La conversión de decimal a hexadecimal, y viceversa, se realiza de la misma forma que en el programa n.º 6 por medio de funciones de manipulación de cadenas.

LISTADO DEL PROGRAMA

```
20 CLS
30 PRINT "DIRECCIONAMIENTO REL. (8 BITS)"
40 PRINT
45 PRINT "INTROD. DIREC. ""FROM"" EN HEX?";
50 INPUT C$
60 PRINT C$
70 GOSUB 330
80 IF F=0 THEN GOTO 110
```

```
 90 PRINT "ERROR, REINTRODUCIR"
100 GOTO 40
110 LET J=X
120 PRINT "INTROD. DIREC. ""TO"" EN HEX?";
130 INPUT C$
140 PRINT C$
150 GOSUB 330
160 IF F=0 THEN GOTO 190
170 PRINT "ERROR, REINTRODUCIR"
180 GOTO 120
190 LET D=X-J
200 IF D<=129 AND D>=-126 THEN GOTO 230
210 PRINT D;"(DEC) OFFSET DEM. GRANDE,REINT."
220 GOTO 40
230 IF D<2 THEN GOTO 260
240 LET Z=D-2
245 LET H$="FORWARD"
250 GOTO 270
260 LET Z=254+D
265 LET H$="ATRAS O EN LUGAR"
270 LET G=INT (Z/16)
280 LET E$=CHR$ (G+28)
290 LET W$=CHR$ (Z-16*G+28)
300 PRINT "OFFSET "; H$;"=";E$; W$
320 GOTO 40
330 LEF F=0
340 LET X=0
350 LET L=LEN C$
360 FOR I=0 TO L-1
370 LET R=CODE C$(L-I)-28
380 IF R<0 OR R>15 THEN LET F=1
390 LET X=X+R*16**I
400 NEXT I
410 RETURN
```

TRAZADOS GENERADOS POR COMPUTADORA

9. Alisado de curvas e interpolación/extrapolación con polinomios de Lagrange

DESCRIPCION DEL PROGRAMA

Este programa es el equivalente de una «curva francesa». Su aplicación principal es la interpolación y extrapolación de resultados medidos, dadas X(i), Y(i) (i = 1,..., n), el programa determina un polinomio único de grado (n – 1) que se ajusta a estos *n* puntos y evalúa el polinomio para valores arbitrarios de la variable independiente X. El método difiere de la aproximación de los mínimos cuadrados en que el polinomio resultante se ajusta exactamente a los puntos X(i), Y(i).

La siguiente fórmula, atribuida al matemático francés Lagrange, describe el polinomio

Fórmula 9-1

$$y(x) = \sum_{i=1}^{n} y_i \times \frac{\prod_{s=1,\, s \neq i}^{n} (x - x_s)}{\prod_{s=1,\, s \neq i}^{n} (x_i - x_s)}$$

Para aplicaciones prácticas es recomendable tener cuidado en seleccionar solamente puntos contiguos para calcular el polinomio de modo que no se obtengan curvas con bruscas oscilaciones como extrapolaciones e interpolaciones de funciones relativamente «suaves». Por regla general, un polinomio de tercer grado, basado en cuatro puntos cercanos (n = 4), será satisfactorio.

Una característica importante del programa es que una «ventana» de interpolación móvil puede establecerse de modo que sólo k valores medios (k < = n) más próximos al punto de interpolación se incluirán en el cálculo.

INSTRUCCIONES

Al pulsar la tecla RUN se inicializa el programa y emite llamadas orientativas correspondientes al número de puntos (n < = 20) y los sucesivos valores X(i) e Y(i) de puntos en orden creciente de X, el número de puntos en la «ventana» y el valor de X en el que la función debe interpolarse o extrapolarse. La siguiente opción es recalcular la función para otra X, cambiar la anchura de la «ventana» o iniciar una nueva ejecución.

EJEMPLO

Enunciado del problema

Como se muestra en la figura 9-1, se han medido seis puntos. Suponer una anchura de «ventana» de cuatro y evaluar la función en X = 2.1 (interpolación) y en X = 7.0 (extrapolación). A continuación, cambiar la anchura de «ventana» a cinco puntos y volver a calcular los mismos dos puntos.

Ejemplo de ejecución

```
INTERPOLACION DE LAGRANGE
NO. PUNTOS?6
INTRODUCIR X(1)?-0.5
INTRODUCIR Y(1)?0
INTRODUCIR X(2)?0.5
INTRODUCIR Y(2)?0.7
INTRODUCIR X(3)?2
INTRODUCIR Y(3)?2
INTRODUCIR X(4)?3
INTRODUCIR Y(4)?2.6
INTRODUCIR X(5)?4.5
INTRODUCIR Y(5)?2.8
INTRODUCIR X(6)?6
INTRODUCIR Y(6)?3.3
```

```
NO. PUNTOS EN VENTANA?4
CALCULAR Y(X),X=?2.3
Y(2.3)=2.20996

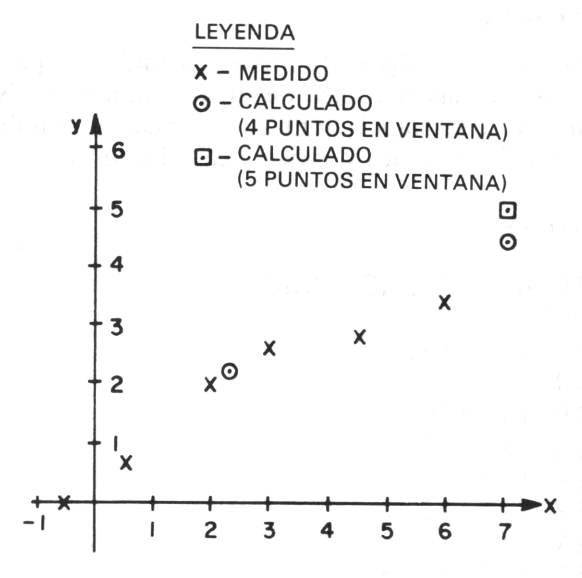
NUEVA X 1,NUE.VENT2,NUE.EJEC 3?1
CALCULAR Y(X), X=?7
Y(7)=4.4333333

NUEVA X 1,NUE.VENT2,NUE.EJEC 3?2
NO. PUNTOS EN VENTANA?5
CALCULAR Y(X), X=?2.3
Y(2.3)=2.22256

NUEVA X 1,NUE.VENT 2,NUE.EJEC 3?1
CALCULAR Y(X), X=?7
Y(7)=5.1909091

NUEVA X 1,NUE.VENT2,NUE.EJEC 3?1
CALCULAR Y(X), X=?4.5
Y(4.5)=2.8
```

Figura 9-1



Discusión de resultados

El ejemplo muestra que la anchura de «ventana» afecta mucho a los valores extrapolares. El valor interpolado cambió solamente desde 2.20... a

2.22..., mientras que el valor extrapolado cambió desde 4.43... a 5.19... El usuario ha de decidir qué anchura de «ventana» es más adecuada. El ejemplo muestra también que el polinomio de Lagrange se ajusta exactamente a cada punto dado. Cuando Y(4.5) se calcula por el programa, genera 2.8, que es el valor exacto originalmente introducido para Y(4.5).

OBSERVACIONES SOBRE LA PROGRAMACION

El programa comprueba las entradas correctas en las líneas 80, 170 y 280. La bifurcación se realiza en tres sentencias simples en las líneas 560-580. La evaluación de la fórmula de Lagrange tiene lugar en un lazo doble en las líneas 430-500.

LISTADO DEL PROGRAMA

```
  5 CLS
 20 DIM X(20)
 30 DIM Y(20)
 40 PRINT "INTERPOLACION DE LAGRANGE"
 50 PRINT "NO. PUNTOS?";
 60 INPUT A
 70 PRINT A
 80 IF A<=20 AND A 2>AND A=INT ABS A THEN
    GOTO 110
 90 PRINT "ERROR"
100 GOTO 50
110 FOR I=1 TO A
120 PRINT "INTRODUCIR X(";I;")?";
130 INPUT X
140 PRINT X
150 LET X(I)=X
160 IF I=1 THEN GOTO 200
170 IF X(I)>X(I-1) THEN GOTO 200
180 PRINT "X(I+1)<=X(I)"
190 GOTO 120
200 PRINT "INTRODUCIR Y(";I;")?";
210 INPUT Y
220 PRINT Y
230 LET Y(I)=Y
240 NEXT I
250 PRINT "NO. PUNTOS EN VENTANA?";
260 INPUT B
270 PRINT B
```

```
280 IF B<=A AND B>1 AND B=INT ABS B THEN
    GOTO 310
290 PRINT "ERROR"
300 GOTO 250
310 LET C=INT (B/2)
320 LET I=1
330 PRINT "CALCULAR Y(X), X=?";
340 INPUT W
350 PRINT W
360 IF W<=X(I) OR I=A THEN GOTO 390
370 LET I=I+1
380 GOTO 360
390 LET Z=0
400 LET D=I-C
410 IF D<1 THEN LET D=1
420 IF I-C+B>A THEN LET D=A-B+1
430 FOR I=D TO D+B-1
440 LET E=1
450 FOR J=D TO D+B-1
460 IF J=I THEN GOTO 480
470 LET E=E*(W-X(J))/(X(I)-X(J))
480 NEXT J
490 LET Z=Z+E*Y(I)
500 NEXT I
510 PRINT "Y(";W;")=";Z
520 PRINT
530 PRINT "NUEVA X 1, NUE.VENT.2, NUE.EJEC.3?";
540 INPUT F
550 PRINT F
560 IF F=1 THEN GOTO 310
570 IF F=2 THEN GOTO 250
580 IF F=3 THEN GOTO 5
590 GOTO 530
```

10. Escalas legibles para trazados por computadora

DESCRIPCION DEL PROGRAMA

Todo buen programa de trazado debe tener la capacidad adecuada para calcular automáticamente las escalas para los ejes X, Y o Z. Si los datos a trazar se generan por la computadora, el usuario no podrá prever siempre las escalas necesarias. Sin embargo, el Timex/Sinclair 1000 puede automatizar la tarea y, además, hacerlo de forma inteligente, para conseguir escalas «legibles». El programa, dado el margen de una variable, produce una escala lineal leible con intervarlos uniformes. La salida del programa puede utilizarse, entonces, para controlar cualquier rutina de trazado. Una escala lineal leible se define aquí como una escala con anchura de intervalo igual al producto de una potencia entera de 10 y 1, 2 ó 5 y valores de escala que son múltiples enteros de la anchura del intervalo.

La anterior definición permite valores de escala tales como:

– 0.5, 0.0, 0.5, 1.0,...
1.24, 1.26, 1.28,...
100, 200, 300,..., etc.,

pero prohibiría los ejemplos siguientes:

– 1, 4, 9,...
1.2, 1.31, 1.42,...
0, 4, 8,..., etc.

Dado un mínimo MIN, un máximo MAX de la matriz a trazar y un número aproximado de intervalos de escala deseados N, el programa calcula cuatro parámetros: un nuevo mínimo MINP, un número máximo MAXP, una anchura de intervalo DIST y varios intervalos de escala NP (que ajustarán mejor el trazado y aproximará N. Los parámetros calculados, además de producir una escala legible, satisfarán las desigualdades siguientes:

$(\mathrm{MIN} - \mathrm{DIST}) < \mathrm{MINP} <= \mathrm{MIN},$
$\mathrm{MAX} <= \mathrm{MAXP} < (\mathrm{MAX} + \mathrm{DIST})$
$\mathrm{N}/\sqrt{2.5} < \mathrm{NP} < (\mathrm{N}\sqrt{2.5} + 2).$

Una característica singular del programa es que introduce una puerta estrecha $\pm$ J (línea 200) alrededor de los valores MIN y MAX para evitar

un margen innecesariamente grande entre MINP y MAXP debido al redondeo de la computadora.

INSTRUCCIONES

Inicializar el programa pulsando la tecla RUN; a continuación seguir las indicaciones para el mínimo y el máximo de su matriz y el número aproximado deseado de intervalos de escala. El programa proporciona, entonces, un nuevo mínimo, un nuevo máximo, las dimensiones del intervalo y el número de intervalos.

EJEMPLOS

Enunciado del problema

Calcular los valores de escala para una matriz con MIN, MAX y N, respectivamente iguales a:

1. – 3.1, 11.1, 5
2. 345, 510, 20
3. – 1100, – 520, 50
4. 17.5, 17.9, 10

Ejemplo de ejecución

```
PROGRAMA CONSTRUCCION ESCALAS

INTRODUCIR MINIMO?-3.1
INTRODUCIR MAXIMO?11.1
INT. NO. APROX. DE INTERVALOS?5
NUEVO MIN=          -4
NUEVO MAX=          12
NUEVO INTERVALO=    2
NO. INTERVALOS=     8

INTRODUCIR MINIMO?345
INTRODUCIR MAXIMO?510
INT. NO. APROX. DE INTERVALOS?20
NUEVO MIN=          340
NUEVO MAX=          510
NUEVO INTERVALO=    10
NO. INTERVALOS=     17
```

```
INTRODUCIR MINIMO?-1100
INTRODUCIR MAXIMO?-520
INT. NO. APROX. DE INTERVALOS?50

NUEVO MIN=          -1100
NUEVO MAX=          -520
NUEVO INTERVALO=    10
NO. INTERVALOS=     58

INTRODUCIR MINIMO?17.9
INTRODUCIR MAXIMO?17.5
ERROR, MIN>=MAX

INTRODUCIR MINIMO?17.5
INTRODUCIR MAXIMO?17.9
INT. NO. APROX. DE INTERVALOS?10
NUEVO MIN=          17.5
NUEVO MAX=          17.9
NUEVO INTERVALO=    .05
NO. INTERVALOS=     8
```

Discusión de los resultados

Los cuatro ejemplos indican que el programa produce escalas «legibles» para una amplia gama de entradas. El primer ejemplo daría los valores de escala de – 4, – 2, 0, 2, ..., 12. El segundo ejemplo proporcionaría 340, 350, ..., 510. El tercer ejemplo daría como resultado los valores – 1100, – 1090, ..., – 520. El cuarto ejemplo proporcionaría 17.5, 17.55,..., 17.9. El cuarto ejemplo muestra también el mensaje de error cuando el mínimo y el máximo se introdujeron por error en orden inverso.

OBSERVACIONES SOBRE LA PROGRAMACION

La comprobación de errores se produce en la línea 100. La determinación de las escalas se realiza en las líneas 240-270. La sentencia PRINT en la línea 340 realiza también el cálculo para ahorrar código.

LISTADO DEL PROGRAMA

```
20 CLS
30 PRINT "PROGRAMA CONSTRUCCION ESCALAS"
40 PRINT
```

```
 45 PRINT "INTRODUCIR MINIMO?";
 50 INPUT X
 60 PRINT X
 70 PRINT "INTRODUCIR MAXIMO?";
 80 INPUT Y
 90 PRINT Y
100 IF Y>X THEN GOTO 130
110 PRINT "ERROR, MIN>=MAX"
120 GOTO 40
130 PRINT "INT. NO. APROX. DE INTERVALOS?";
140 INPUT N
150 PRINT N
160 IF N>0 AND N=INT ABS N THEN GOTO 190
170 PRINT "ERROR"
180 GOTO 130
190 LET D=(Y-X)/N
200 LET J=D/1E5
210 LET E=INT (LN D/LN 10)
220 LET F=D/10**E
230 LET V=10
240 IF F<SQR 50 THEN LET V=5
250 IF F<SQR 10 THEN LET V=2
260 IF F<SQR 2 THEN LET V=1
270 LET C=V*10**E
280 LET G=INT (X/C)
290 IF ABS (G+1-X/C)<J THEN LET G=G+1
300 LET A=C*G
310 LET H=INT (Y/C)+1
320 IF ABS (Y/C+1-H)<J THEN LET H=H-1
330 LET B=C*H
340 PRINT "NUEVO MIN=",A,"NUEVO MAX=",B,
    "NUEVO INTERVALO=",
345 PRINT C,"NO. INTERVALOS=",H-G
360 GOTO 40
```

PROBABILIDAD Y ESTADISTICA

11. Combinaciones, permutaciones y factoriales

DESCRIPCION DEL PROGRAMA

Este programa evalúa las tres funciones más importantes en estadística y en combinatoria.

Combinaciones. El programa calcula el número de combinaciones posibles de N objetos diferentes tomados de un conjunto de M objetos. Al cambiar la secuencia de los objetos en una combinación no se obtiene una nueva combinación.

La función resultante C(M,N) se denomina también el coeficiente binómico y puede representarse como sigue:

Fórmula 11-1

$$C\ (M,N) = \binom{M}{N} = \frac{M!}{(M-N)!\ N!} =$$

$$= \left(\frac{M+1}{N} - 1\right)\left(\frac{M+1}{N-1} - 1\right)\dots\left(\frac{M+1}{1} - 1\right)$$

La primera expresión, que es la definición del coeficiente binómico C(M,N) puede dar lugar a grandes errores al dividir un gran número por otro también grande. Si alguna de las factoriales en el numerador o en el denominador es mayor que 33!, entonces, no se podría evaluar la primera expresión por el Timex/Sinclair 1000. Por el contrario, la segunda expresión evita el problema del desbordamiento de la capacidad, por lo que será la que utilizaremos para evaluar C(M,N).

Permutaciones. El programa calcula el número de permutaciones posibles con M objetos tomados N a N. Al cambiar el orden de los objetos en una permutación se obtiene una permutación diferente.

La función resultante P(M,N) se calcula como sigue:

Fórmula 11-2

$$P(M,N) = M(M-1)\ldots(M-N+1)$$

Factoriales. Una factorial F(M) es igual al número de formas en las que M objetos puede redisponerse. La función se define y calcula como sigue:

Fórmula 11-3

$$F(M) = M(M-1)\ldots 1$$

EJEMPLOS

Enunciados de problemas

1. ¿En cuántas formas pueden 10 personas disponerse en una fila? Evaluar F(10).
2. ¿Cuántos grupos de 3 personas pueden seleccionarse de entre un grupo de 20 cuando importa el orden en cada grupo de 3? Evaluar P(20,3).
3. ¿Cuántos grupos de 3 pueden seleccionarse de entre 20 personas, cuando el orden no importa? Evaluar C(20,3).
4. Lo mismo que en el ejemplo 3 pero seleccionando un grupo de 17 personas de entre 20. Evaluar C(20,17).

Ejemplo de ejecución

```
FACTORIALES 1,PERMUTACIONES 2
COMBINACIONES (COEF. BIN.) 3?1
FACTORIAL DE?10
FACT(10)=3628800

FACTORIALES 1,PERMUTACIONES 2
COMBINACIONES (COEF. BIN.) 3?2
M=?20
N=?3
PERM(20,3)=6840
```

```
FACTORIALES 1,PERMUTACIONES 2
COMBINACIONES (COEF. BIN.) 3?3
M=?20
N=?3
COMB(20,3)=1140

FACTORIALES 1,PERMUTACIONES 2
COMBINACIONES (COEF. BIN.) 3?3
M=?20
N=?17
COMB(20,17)=1140

FACTORIALES 1,PERMUTACIONES 2
COMBINACIONES (COEF. BIN.) 3?
```

Discusión de resultados

El ejemplo 1 produce un número muy grande, los ejemplos 3 y 4 proporcionan el mismo resultado. La razón para la igualdad es que cuando se elige un grupo de 17, el grupo restante será tres, que es lo mismo que en el ejemplo 3.

OBSERVACIONES SOBRE LA PROGRAMACION

La comprobación de los errores de entrada se realiza en las líneas 60 y 130. La misma sentencia PRINT en la línea 450 se utiliza para las permutaciones y las combinaciones asignando un nombre adecuado a la variable C$. El programa hace uso de la igualdad C(M,N) = C(M,M – N) y selecciona en la línea 250 el recorrido de cálculo más rápido.

LISTADO DEL PROGRAMA

```
 20 CLS
 30 PRINT
 35 PRINT "FACTORIALES 1,PERMUTACIONES 2"
 37 PRINT "COMBINACIONES (COEF. BIN.) 3?";
 40 INPUT F
 50 PRINT F
 60 IF F<>1 AND F<>2 AND F<>3 THEN GOTO 30
 70 GOTO 100*F
100 PRINT "FACTORIAL DE?";
110 INPUT M
120 PRINT M
```

```
130 IF M<=0 OR M<>INT ABS M OR M>33
    THEN GOTO 100
140 LET B=1
150 FOR I=1 TO M
160 LET B=B*I
170 NEXT I
180 PRINT "FACT(";M;")=";B
190 GOTO 30
200 LET C$="PERM("
210 GOSUB 480
220 IF M>=N THEN GOTO 250
230 LET B=0
240 GOTO 450
250 LET B=1
260 FOR I=M-N+1 TO M
270 LET B=B*I
280 NEXT I
290 GOTO 450
300 LET C$="COMB("
310 GOSUB 480
320 IF M>=N THEN GOTO 350
330 LET B=0
340 GOTO 450
350 IF M<>N THEN GOTO 380
360 LET B=1
370 GOTO 450
380 LET Y=N
390 IF M<2*N THEN LET Y=M-N
400 LET K=M+1
410 LET B=1
420 FOR I=1 TO Y
430 LET B=B*(K/I-1)
440 NEXT I
450 PRINT C$;M;",";N;")=";B
470 GOTO 30
480 PRINT "M=?";
490 INPUT M
495 PRINT M
500 PRINT "N=?";
510 INPUT N
515 PRINT N
520 IF M=INT ABS M AND N=INT ABS N AND M*N<>0
    THEN RETURN
530 PRINT "ERROR"
540 GOTO 480
```

12. La función de error ERF y su función ERFC complementaria

DESCRIPCION DEL PROGRAMA

Los atributos de muchos objetos en la naturaleza, por ejemplo, la altura en pulgadas de la población masculina de una cierta ciudad, o los coeficientes de inteligencia (IQ) de la población, siguen estrictamente la distribución normal o gausiana (la bien conocida curva de la campana de Gauss). Una distribución normal particular se expresa completamente mediante dos parámetros, la media y la desviación cuadrática media. La función de error (ERF) es igual a la distribución normal acumulativa y la función de error complementaria (ERFC) es igual a 1-ERF según las definiciones siguientes:

Fórmula 12-1

$$\mathrm{Erf}\ (\mu,\sigma,x) = \frac{1}{\sigma\sqrt{2\pi}} \int_{-\infty}^{x} e^{-\frac{1}{2}\left(\frac{t-\mu}{\sigma}\right)^2} dt \qquad \mathrm{Erfc} = 1 - \mathrm{Erf}$$

Las anteriores fórmulas no pueden evaluarse de forma cerrada. El método utilizado en este programa para calcular ERF y ERFC es iterativo y lleva a resultados que son correctos dentro de la capacidad de visualización completa del Timex/Sinclair 1000 para ambas funciones. Por comparación, el método normal utilizado para calcular ERF y ERFC se basa en la expansión polinómica de un quinto o sexto grado. Y es preciso para solamente 3 a 4 lugares decimales. La ERFC calculada mediante expansión polinómica pierde completamente la exactitud en los extremos, o colas, de la distribución.

INSTRUCCIONES

Al pulsar la tecla RUN se emiten mensajes de petición de datos para la media de distribución, desviación standard y el límite de distribución superior en donde deben evaluarse ERF y ERFC.

EJEMPLOS

Enunciados de problemas

1. Dada una población con un coeficiente de inteligencia (IQ) medio de 100 y una desviación cuadrática media (desviación estándar) de 15, ¿Qué fracción de la población tendrá valores IQ mayores que 135?

2. El peso medio de un saco de fertilizante es de 45 libras. El proceso de fabricación introduce una desviación cuadrática media de 1,5 libras. ¿Qué proporción de sacos será más pesada que 49,5 libras?

3. Probablemente haya oído hablar acerca del límite de sigma de 2,33 (sigma es la desviación cuadrática media). ¿Qué porcentaje de población incluye?

Ejemplo de ejecución

```
PROGRAMA DE FUNCION DE ERROR
DISTR. MEDIA=?100
DESVIACION EST.=?15
LIMITE DISTR. SUPERIOR=?135
ERF=0.99018467  ERFC=.0098153287

DISTR. MEDIA=?45
DESVIACION ST.=?1.5
LIMITE DISTR. SUPERIOR=?49.5
ERF=0.9986501   ERFC=.001349898

DISTR. MEDIA=?100
DESVIACION ST.=?10
LIMITE DISTR. SUPERIOR=?123.3
ERF=0.99009693  ERFC=.0099030756
```

Discusión de los resultados

Solamente el 0,98% de la población tendrá IQ mayores que 135. El segundo ejemplo indica que el peligro de sacos pesados es bastante pequeño, solamente del 0,13%. El tercer ejemplo indica que la desviación estándar de 2,33 incluye al 95% de la población.

OBSERVACIONES SOBRE LA PROGRAMACION

Dependiendo del valor del argumento, las funciones se evalúan como fracciones continuas con un número determinado de elementos o se evalúan de forma iterativa hasta que convergen.

LISTADO DEL PROGRAMA

```
 20 CLS
 30 PRINT
 35 PRINT "PROGRAMA DE FUNCION DE ERROR"
 40 PRINT "DISTR. MEDIA=?";
 50 INPUT B
 60 PRINT B
 70 PRINT "DESVIACION EST.=?";
 80 INPUT C
 90 PRINT C
100 IF C>=0 THEN GOTO 130
110 PRINT "ERROR"
120 GOTO 70
130 PRINT "LIMITE DISTR. SUPERIOR=?";
140 INPUT D
150 PRINT D
160 LET X=(D-B)/C
170 LET X=X/SQR 2
180 LET Y=1/SQR PI*EXP (-X*X)
190 IF X<SQR 2 THEN GOTO 280
200 LET A=14/X
210 FOR I=27 TO 1 STEP -1
220 LET A=I/2/(X+A)
230 NEXT I
240 LET Z=Y/(X+A)
250 LET Z=Z/2
260 LET N=1-Z
270 GOTO 390
280 LET I=1
290 LET T=2*X*Y
300 LET N=T
310 LET T=2*X*X*T/(2*I+1)
320 LET M=N+T
330 IF M=N THEN GOTO 370
340 LET N=M
350 LET I=I+1
360 GOTO 310
370 LET N=N/2+.5
380 LET Z=1-N
390 PRINT "ERF=";N,"ERFC=";Z
400 PRINT
410 GOTO 40
```

13. Distribuciones binomia e hipergeométrica

DESCRIPCION DEL PROGRAMA

Estas dos distribuciones se producen en muchos problemas que se refieren a la teoría del muestreo, al control de calidad y a los juegos. La distribución binomia se aplica al caso de un universo estadístico grande, mientras que la distribución hipergeométrica se aplica a casos de un universo pequeño, en donde las muestras se toman sin ninguna sustitución. Por ejemplo, la probabilidad de encontrar X elementos defectuosos en una muestra a partir de un universo grande con una proporción conocida de elementos defectuosos, seguiría la distribución binomia. Análogamente, la probabilidad de encontrar exactamente dos bolas rojas en una muestra de cinco bolas tomadas todas a la vez a partir de universo de diez bolas, tres de las cuales son rojas, seguiría la distribución hipergeométrica. Ambas distribuciones son muy útiles para estimar acontecimientos de la vida real. Las siguientes fórmulas se aplican para las dos distribuciones.

A. Distribución binomia

Dado un universo grande con una proporción de P elementos especiales (defectuosos, marcados, etc.) y un tamaño de muestra de N, la probabilidad de encontrar X elementos especiales en esa muestra es como sigue:

Fórmula 13-1

$$\text{Prob } (N,P,X) = \binom{N}{X} p^{x} (1 - p)^{N-X}$$

B. Distribución hipergeométrica.

Dado un universo (población estadística) de pequeña magnitud de tamaño M con K elementos defectuosos, la probabilidad de encontrar X elementos defectuosos en una muestra de tamaño N es como sigue:

Fórmula 13-2

$$\text{Prob } (M,N,K,X) = \frac{\binom{K}{X}\binom{M - K}{N - X}}{\binom{M}{N}}$$

INSTRUCCIONES

El primer mensaje orientativo después de pulsar la tecla RUN proporciona la elección de la distribución binomia (1) o hipergeométrica (2). Si se selecciona la distribución binomia, sigue una llamada orientativa para el tamaño de la muestra N y para la proporción de elementos especiales P en el universo. En este y siguientes ejemplos, cuando dicha llamada orientativa exija más de un valor de entrada, ha de introducirse por el teclado cada valor por separado seguido por «Enter» (introducir). La siguiente llamada corresponde al número de elementos especiales X en la muestra, para lo que debe calcularse la probabilidad. Después de visualizar PROB(X) y la probabilidad acumulativa, la siguiente llamada orientativa da una elección entre una nueva X o una nueva ejecución.

Análogamente, para la distribución hipergeométrica, la secuencia de llamadas orientativas es la misma, con la excepción de que se requiere la introducción de tres parámetros M, K y N (tamaño del universo, número de elementos especiales y el tamaño de la muestra) para describir la distribución.

EJEMPLOS

Enunciados de problemas

1. **Distribución binomia.** Se sabe que el 4% de la marca de un fabricante es de elementos defectuosos. ¿Cuál es la probabilidad de encontrar, a lo sumo, un elemento defectuoso en una muestra de cinco?

2. **Distribución hipergeométrica.** En una caja de 20 chocolates, 5 son oscuros y 15 son claros. ¿Cuál es la probabilidad de encontrar uno o más chocolates oscuros en un puñado de tres?

Ejecuciones de ejemplos

```
BINOMIA 1,HIPERGEOMETRICA 2?1
INTRODUCIR N,P?5         .04
X=?1
PROB(1)=0.16986931
SUMA DE P(X)=            0.16986931
NUEVA X 1,NUEVA EJEC. 2?1
X=?0
PROB(0)=0.8153727
SUMA DE P(X)=            0 98524201
NUEVA X 1, NUEVA EJEC. 2?2

BINOMIA 1,HIPERGEOMETRICA 2?2
INTRODUCIR M,K,N         20
```

```
5                  3
X=?1
PROB(20,5,3,1)=      0.46052632
SUMA DE (X)=0.46052632

NUEVA X 1,NUEVA DISTR. 2?1
X=?2
PROB(20,5,3,2)=      0.13157895
SUMA DE (X)=0.59210526
NUEVA X 1,NUEVA DISTR. 2?1
X=?3
PROB(20,5,3,3)=      .0087719298
SUMA DE(X)=0.60087719
NUEVA X 1,NUEVA DISTR. 2?1
X=?0
PROB(20,5,3,0)=0.39912281
SUMA DE(X)=1
NUEVA X 1,NUEVA DISTR. 2?
```

Discusión de los resultados

En el primer ejemplo, la probabilidad de encontrar un elemento defectuoso en una muestra es 0.169..., y de encontrar uno sin defecto es 0.815... La probabilidad de encontrar 0 ó 1 elementos defectuosos en una muestra de 5 es, pues, P(0) + P(1) = 0,985...

En el segundo ejemplo, la probabilidad de encontrar uno o más chocolates oscuros en una muestra de tamaño tres es la suma de probabilidades de encontrar uno, dos o tres y es igual a 0.6. El mismo resultado puede obtenerse más fácilmente calculando la probabilidad de ningún chocolate oscuro y restándola de 1.0, como se muestra en la última ejecución del ejemplo.

OBSERVACIONES SOBRE LA PROGRAMACION

Una comprobación amplia de los valores de entrada se realiza en las líneas 90, 150, 190, 280, 360 y 400. Los coeficientes binomios se calculan en la subrutina entre las líneas 520 y 620 de la misma forma que en el programa 11. Obsérvese el empleo de sentencias PRINT en las líneas 230 y 460, que incluyen la reimpresión de parámetros de entrada para una más fácil interpretación de los resultados.

LISTADO DEL PROGRAMA

```
 20 CLS
 40 PRINT
 41 PRINT "BINOMIA 1,HIPERGEOMETRICA 2?";
 45 LET Z=0
 50 INPUT W
 60 PRINT W
 70 IF W=1 THEN GOTO 100
 80 IF W=2 THEN GOTO 290
 90 GOTO 40
100 PRINT "INTRODUCIR N,P?";
110 INPUT N
120 PRINT N,
130 INPUT P
140 PRINT P
150 IF N<>INT N OR N<=0 OR P<=0 OR P>=1
    THEN GOTO 100
160 PRINT "X=?";
170 INPUT X
180 PRINT X
190 IF X<>INT X OR X<0 OR X>N THEN GOTO 160
200 GOSUB 520
210 LET T=B*P**X*(1-P)**(N-X)
220 LET Z=Z+T
230 PRINT "PROB(";X;")=";T,"SUM DE P(X)=",Z
240 PRINT "NUEVA X 1,NUEVA EJEC.?2";
250 INPUT W
260 PRINT W
270 IF W=1 THEN GOTO 160
280 GOTO 40
290 PRINT "INTRODUCIR M,K,N",
300 INPUT E
310 PRINT E,
320 INPUT G
330 PRINT G,
340 INPUT F
350 PRINT F
360 IF E<>INT E OR G<>INT G OR F<>INT F
    THEN GOTO 290
365 IF E<=0 OR F<=0 OR G<=0 OR F>E OR G>E
    THEN GOTO 290
370 PRINT "X=?";
380 INPUT D
390 PRINT D
```

```
400 IF D<>INT D OR D<0 OR D>G THEN GOTO 370
410 LET N=G
420 LET X=D
422 GOSUB 520
424 LET J=B
426 LET N=E-G
428 LET X=F-D
430 GOSUB 520
432 LET J=J*B
434 LET N=E
436 LET X=F
438 GOSUB 520
440 LET J=J/B
450 LET Z=Z+J
460 PRINT "PROB(";E;",";G;",";
465 PRINT F;",";D;")=",J,"SUM DE(X)=";Z
470 PRINT "NUEVA X 1,NUEVA DISTR. 2?";
480 INPUT W
490 PRINT W
500 IF W=1 THEN GOTO 370
510 GOTO 40
520 LET Y=X
530 IF X>(N-X) THEN LET Y=N-X
540 IF Y<>0 THEN GOTO 570
550 LET B=1
560 RETURN
570 LET B=1
580 LET L=N+1
590 FOR I=1 TO Y
600 LET B=B*(L/I-1)
610 NEXT I
620 RETURN
```

14. Desviantes pseudo-aleatorias (gaussianas) normales y uniformes

DESCRIPCION DEL PROGRAMA

Este programa produce una secuencia de desviantes aleatorias (gaussianas) normalmente distribuidas o uniformes. Dichas secuencias suelen utilizarse para la realización de modelos con el método de Monte Carlo, en donde no sea factible un cálculo directo.

Un ejemplo de una aplicación para la que no es factible un cálculo exacto es el de la satisfacción de los abonados telefónicos. La satisfacción de un grupo de abonados telefónicos depende de la distribución probabilística de la sonoridad de quienes hablan, de la distribución del ruido y de la distribución de pérdidas de transmisión en una conexión telefónica. Aunque cada uso de estas distribuciones puede estimarse como gaussiana con una desviación estándar y media conocida a partir de encuestas, la determinación de la distribución combinada es difícil desde el punto de vista del cálculo. Sin embargo, el efecto combinado de todos estos factores puede simularse fácilmente generado, de forma repetida, desviantes pseudo-aleatorias a partir de cada una de estas distribuciones y multiplicando las probabilidades individuales.

Este programa actuaría normalmente como una subrutina en una aplicación proporcionada por el usuario. Las fórmulas utilizadas para generar las desviantes pseudo-aleatorias normales y uniformes son las siguientes:

A. Distribución uniforme

Fórmula 14-1

$$X_i = \text{Fract}(997\ X_{i-1} + \pi)$$

en donde RND es la función pseudo-aleatoria del Timex/Sinclair 1000 BASIC y X e Y son los límites inferior y superior de la variable aleatoria. La anterior fórmula genera los valores de distribución uniforme entre los límites inferior y superior X e Y.

B. Distribución normal (gaussiana)

Fórmula 14-2

$X_i = (-2 \ln U_1)^{1/2} \sin (2\pi U_2)$

en donde U_1 y U_2 son dos desviantes pseudo-aleatorias *uniformemente* distribuidas, y secuencialmente generadas, entre 0 y 1. Las desviaciones pseudo-aleatatorias *normalmente* distribuidas resultantes tienen una media de 0 y una desviación estándar de 1. Los valores se transforman, entonces, por el programa en una distribución con una media arbitraria M y una desviación estándar S multiplicando cada desviante aleatoria con S y añadiendo luego M.

INSTRUCCIONES

Pulsar la tecla RUN para iniciar el programa y luego, seguir las llamadas orientativas para desviantes uniformes (1) o gaussianas (2). La siguiente llamada orientativa es para el elemento de arranque que inicializa la función aleatoria (RND) del Timex/Sinclair 1000. El valor del elemento de arranque debe estar comprendido entre 0 y 65535. Al introducir 0 se iniciará siempre una secuencia aleatoria diferente; de no ser así, la secuencia se determinará previamente. A continuación, seguir las llamadas orientativas para los límites inferior y superior para la distribución uniforme o para la desviación estándar o media de la distribución gaussiana, respectivamente.

EJEMPLOS

Enunciados de problemas

1. Generar varias desviantes uniformemente distribuidas entre 10 y 20.
2. Generar desviantes gaussianas con una media de 100 y una desviación estándar de 10. Una posible interpretación de los valores resultantes de las desviantes gaussianas podría ser una lectura aleatoria de los cocientes intelectuales (IQ).

Ejemplos de ejecución

```
UNIFORME 1, GAUSSIANA 2?1
ELEMENTO ARRANQUE=?123
LIMITE INFERIOR, SUPERIOR?10,20
DESVIANTE ALEATORIA=11.418915

MAS VALORES 1, NUEVA EJEC. 2?1
DESVIANTE ALEATORIA=16.428375
```

```
MAS VALORES 1, NUEVA EJEC. 2?1
DESVIANTE ALEATORIA=12.132111

MAS VALORES 1, NUEVA EJEC. 2?1
DESVIANTE ALEATORIA=19.917297

MAS VALORES 1, NUEVA EJEC. 2?1
DESVIANTE ALEATORIA=13.797302

MAS VALORES 1, NUEVA EJEC. 2?1
DESVIANTE ALEATORIA=14.804688

MAS VALORES 1, NUEVA EJEC. 2?2

UNIFORME 1, GAUSSIANA 2?2
ELEMENTO ARRANQUE=?555
MEDIA=?100
DEV. EST.=?10
DESVIANTE ALEATORIA=90.646835

MAS VALORES 1, NUEVA EJEC. 2?1
DESVIANTE ALEATORIA=114.96507

MAS VALORES 1, NUEVA EJEC. 2?1
DESVIANTE ALEATORIA=108.74587

MAS VALORES 1, NUEVA EJEC. 2?1
DESVIANTE ALEATORIA=116.17717

MAS VALORES 1, NUEVA EJEC. 2?1
DESVIANTE ALEATORIA=100.46652

MAS VALORES 1, NUEVA EJEC. 2?1
DESVIANTE ALEATORIA=114.59181
```

Discusión de los resultados

Las desviantes uniformes están comprendidas entre 10 y 20 y las desviantes gaussianas siguen la distribución normal.

OBSERVACIONES SOBRE LA PROGRAMACION

La comprobación de errores de todas las entradas se realiza en las líneas 130, 220 y 280. El programa muestra el empleo de las funciones RAND y

RND de BASIC. Si el valor del elemento de arranque, o argumento, de RAND se selecciona como 0, una nueva secuencia aleatoria se iniciará cada vez que se ejecute el programa. De cualquier otro modo, los valores pseudo-aleatorios pueden repetirse, si se elige el mismo elemento de arranque, en ejecuciones sucesivas.

LISTADO DEL PROGRAMA

```
 20 CLS
 25 PRINT
 30 PRINT "UNIFORME 1, GAUSSIANA 2?";
 40 INPUT B
 42 PRINT B
 44 PRINT "ELEMENTO ARRANQUE=?;
 46 INPUT S
 48 PRINT S
 50 RAND (S)
 52 IF B=1 THEN GOTO 80
 60 IF B=2 THEN GOTO 230
 70 GOTO 30
 80 PRINT "LIMITE INFERIOR, SUPERIOR?";
 90 INPUT X
100 PRINT X;
110 INPUT Y
120 PRINT ",";Y
130 IF Y>X THEN GOTO 160
140 PRINT "MIN>=MAX"
150 GOTO 80
160 PRINT "DESVIANTE ALEATORIA=";RND*(Y-X)+X
165 PRINT
170 PRINT "MAS VALORES 1, NUEVA EJEC. 2?";
180 INPUT Z
190 PRINT Z
200 IF Z=1 THEN GOTO 160
210 IF Z=2 THEN GOTO 25
220 GOTO 170
230 PRINT "MEDIA=?";
240 INPUT C
250 PRINT C
255 PRINT "DEV. EST.=?";
260 INPUT D
270 PRINT D
280 IF D>=0 THEN GOTO 310
290 PRINT "DEV. EST. < 0"
```

```
300 GOTO 255
310 PRINT "DESVIANTE ALEATORIA=";
315 PRINT SQR (-2*LN RND)*SIN (2*PI*RND)*D+C
320 PRINT
325 PRINT "MAS VALORES 1, NUEVA EJEC. 2?";
330 INPUT Z
340 PRINT Z
350 IF Z=1 THEN GOTO 310
360 IF Z=2 THEN GOTO 25
370 GOTO 320
```

MATEMATICAS

15. Coeficientes Fourier de funciones periódicas

DESCRIPCION DEL PROGRAMA

Una función periódica continua y = f(x) con un período T definido entre – T/2 y + T/2 puede representarse como la suma de términos de las funciones seno y coseno, según las siguientes fórmulas clásicas atribuidas al matemático francés Fourier.

Fórmula 15-1

$$f(x) = \frac{A_o}{2} + \sum_{n=1}^{\infty} (A_n \cos 2n \frac{\pi}{T} x + B_n \sin 2n \frac{\pi}{T} x) = \frac{A_0}{2} + \sum_{n=1}^{\infty} C_n \cos (2n \frac{\pi}{T} x + \theta)$$

$$A_n = \frac{2}{T} \int_{-T/2}^{T/2} f(x) \cos \frac{2n \pi x}{T} dx \qquad C_n = \sqrt{A_n^2 + B_n^2}$$

$$B_n = \frac{2}{T} \int_{-T/2}^{T/2} f(x) \text{sèn} \frac{2n \pi x}{T} dx \qquad \theta = \text{Arctan} (- \frac{Bn}{An})$$

Este programa evalúa las anteriores expresiones realizando una integración numérica a intervalos de muestreo especificados. Si se conoce, por anticipado, que la función es par, f(x) = f(– x), o que es impar, f(x) = – f(– x), el programa puede acelerarse porque sólo han de evaluarse los coeficientes A o B y la integración puede proseguir en solamente una mitad del período T. El programa se aprovecha de esta información.

Aunque el programa solamente visualiza los coeficientes de Fourier sucesivos, estos coeficientes podrían almacenarse fácilmente en el momento

del cálculo en una variable dimensionada (que permita la capacidad de la memoria) para su uso futuro.

INSTRUCCIONES

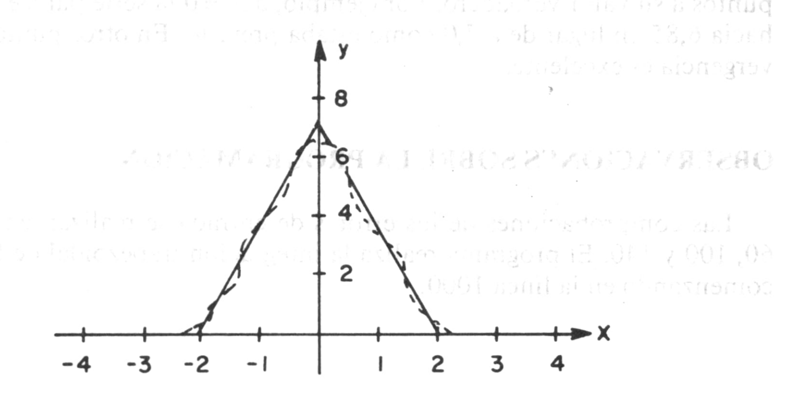
Antes de ejecutar el programa, ha de escribirse una subrutina que defina la función a evaluar comenzando en la línea 1000 y finalizando con RETURN (ver listado del programa y ejemplo). Al pulsar la tecla RUN se inicializa el programa y se emite una llamada orientativa para el tipo de función: (1) para una función par, (2) para una función impar y (3) si no se conoce el tipo de función. Las llamadas orientativas posteriores requieren la magnitud del período T y el número de intervalos de muestreo (dos o más). La selección de un número grande de intervalos proporciona una mayor exactitud pero requiere más tiempo de cálculo. Para una función par o impar, el número requerido de intervalos se aplica a solamente una mitad del período T, con lo que se hacen más pequeños los intervalos y mejor la exactitud del cálculo. Cuando se determina un coeficiente de Fourier, la computadora indicará su valor de forma intermitente. Los sucesivos coeficientes se calculan y visualizan hasta que la pantalla esté llena (error (5) o se pulse la tecla BREAK.

EJEMPLO

Enunciado de problema

La función especificada en las líneas 1000-1020 se muestra en la figura 15-1. Su período está comprendido entre – 3.5 y + 3.5. El programa calcula los coeficientes Fourier sucesivos A0, A1, A2... para esta función par de x. La curva aproximada se muestra a trazos en la misma figura. El cálculo ha de realizarse con 10 y 100 intervalos.

Figura 15-1



Ejemplo de ejecución

```
FUNCION PAR 1, IMPAR 2,
NINGUNA DE ELLAS 3?1
PERIODO T=?7
NO. INTERVALOS=?10
A0=4.025
A1=3.0177238
A2=1.1368327
A3=0.1117739
A4=.070788988
A5=0.14849242
A6=.058580645
A7=.01478557
A8=.027038986
A9=.011989939
A10=0
A11=-.011989941
A12=-.027038986
A13=-.014785568
A14=-.058580644
A15=-0.14849242
A16=-.070788986
A17=-0.1117739
```

```
FUNCION PAR 1, IMPAR 2,
NINGUNA DE ELLAS 3?1
PERIODO T=?7
NO. INTERVALOS=?100
A0=4.00015
A1=3.0345888
A2=1.1793983
A3=0.10390619
A4=.058466219
A5=0.18842451
A6=.08414665
A7=.0001493158
A8=.047250562
A9=.057932911
A10=.0094089601
A11=.0077683475
A12=.03243459
A13=.017811636
A14=.00014726251
A15=.013311727
A16=.018110545
```

Discusión de los resultados

Los coeficientes de Fourier se verificaron de forma independiente con un algoritmo de alta precisión en una computadora diferente; se encontró que con 100 intervalos entre cuatro y cinco lugares decimales son correctos en el Timex/Sinclair 1000. Obsérvese el fenómeno de Gibbs en esquinas agudas, $X = -3.5$, 0, $+3.5$; la expansión de Fourier no converge en esos puntos a su valor verdadero. Por ejemplo, a $X = 0$ la serie parece converger hacia 6,85 en lugar de a 7,0 como estaba previsto. En otros puntos, la convergencia es excelente.

OBSERVACIONES SOBRE LA PROGRAMACION

Las comprobaciones de los errores de entrada se realizan en las líneas 60, 100 y 140. El programa realiza la integración trapezoidal de la función comenzando en la línea 1000.

LISTADO DEL PROGRAMA

```
 30 PRINT "FUNCION PAR 1,IMPAR 2,NINGUNA DE
    ELLAS 3?";
 40 INPUT A
 50 PRINT A
 60 IF A<1 OR A>3 OR A<>INT A THEN GOTO 30
 70 PRINT "PERIODO T=?";
 80 INPUT B
 90 PRINT B
100 IF B<=0 THEN GOTO 70
110 PRINT "NO. INTERVALOS=?";
120 INPUT C
130 PRINT C
140 IF C<2 OR C<>INT C THEN GOTO 110
150 IF A=3 THEN GOTO 500
160 LET D=B/2/C
170 IF A=2 THEN GOTO 340
180 LET J=-1
190 LET J=J+1
195 PAUSE 20
200 LET X=0
210 GOSUB 1000
220 LET E=Y
230 LET F=0
240 LET G=2*J*PI/B
250 FOR I=1 TO C
260 LET X=D*I
270 GOSUB 1000
280 LET F=F+(E+Y)/2*D*COS (G*(X-D/2))
290 LET E=Y
300 NEXT I
310 LET H=4*F/B
320 PRINT "A";J;"=";H
330 GOTO 190
340 LET J=0
350 LET J=J+1
355 PAUSE 20
360 LET X=0
370 GOSUB 1000
380 LET E=Y
390 LET K=0
400 LET G=2*J*PI/B
410 FOR I=1 TO C
420 LET X=D*I
```

```
430 GOSUB 1000
440 LET K=K+(E+Y)/2*D*SIN (G*(X-D/2))
450 LET E=Y
460 NEXT I
470 LET L=4*K/B
480 PRINT "B";J;"=";L
490 GOTO 350
500 LET D=B/C
510 LET J=-1
520 LET J=J+1
525 PAUSE 20
530 LET X=-B/2
540 GOSUB 1000
550 LET E=Y
560 LET K=0
570 LET F=0
580 LET G=2*J*PI/B
590 FOR I=1 TO C
600 LET X=D*I-B/2
610 GOSUB 1000
620 LET M=G*(X-D/2)
630 LET F=F+(E+Y)/2*D*COS M
640 IF J=0 THEN GOTO 660
650 LET K=K+(E+Y)/2*D*SIN M
660 LET E=Y
670 NEXT I
680 LET H=2*F/B
690 PRINT "A";J;"=";H
700 IF J=0 THEN GOTO 520
710 LET L=2*K/B
720 PRINT "B";J;"=";L
730 LET N=SQR (H*H+L*L)
740 LET P=ATN (-L/H)
750 PRINT "C";J;"=";N
760 PRINT "THETA";J;"=";P
770 GOTO 520
1000 LET Y=7-3.5*ABS X
1010 IF ABS X>2 THEN LET Y=0
1020 RETURN
```

16. Soluciones de ecuaciones diferenciales por el método de Runge-Kutta

DESCRIPCION DEL PROGRAMA

Los métodos de Runge-Kutta para soluciones numéricas de ecuaciones diferenciales ordinarias están constituidos por un grupo de fórmulas iterativas que se distinguen por algunas características comunes importantes.

1. La derivada y′ puede darse como una función de x y de y. No ha de proporcionarse como una función explícita de x solamente.
2. El método es de iniciación automática. El conocimiento de la función sólo se requiere en un solo punto para calcular el punto sucesivo en la iteración.
3. El intervalo de integración puede cambiarse cuando prosigue la iteración.
4. Las fórmulas suelen converger rápidamente.

Sucesivamente, fórmulas de Runge-Kutta de más alto orden e intervalos de integración más pequeños dan lugar a soluciones más precisas. El programa evalúa las fórmulas de Runge-Kutta de cuarto orden para resolver ecuaciones diferenciales de primer orden del tipo:

$$y' = f(x,y)$$

En las fórmulas se supone que se conoce f(x,y). La condición inicial de X0 y Y0, el intervalo de integración H a lo largo del eje X y el valor final de X se seleccionan por el usuario. El programa evalúa las fórmulas siguientes:

Fórmula 16-1

$$K_1 = Hf(X_i, Y_i)$$

$$K_2 = Hf(X_i + H/2, Y_i + K_1/2)$$

$$K_3 = Hf(X_i + H/2, Y_i + K_2/2)$$

$$K_4 = Hf(X_i + H/2,\ Y_i + K_3/2)$$

$$Y_{i+1} = Y_i + \frac{1}{6}(K_1 + 2K_2 + 2K_3 + K_4)$$

INSTRUCCIONES

Antes de ejecutar el programa, escribir la función $y' = f(x,y)$ como una subrutina de BASIC que comienza en la línea 1000 y añadirla al programa como se muestra en el listado del programa y en el ejemplo. Utilícese la letra R para la derivada. Al pulsar la tecla RUN se inicia el programa y se produce la llamada orientativa para la magnitud del intervalo H, los valores iniciales de X0 y Y0 y para el valor final de X en el que debe encontrarse la solución. El intervalo H debe ser positivo, aunque X puede ser más pequeña o más grande que X0. No hay otras limitaciones para H. El programa hace que parpadeen los valores x sucesivos a medida que se evalúan y se para cuando se termina el cálculo visualizado Y(X).

EJEMPLO

Enunciado del problema

Dada la ecuación diferencial $y' = x + y$, con la condición inicial $X = 0$ y $Y = 0$. Determinar Y(– 1). Suponer los intervalos de integración de 0,15 y de 0,05.

Ejemplo de ejecución

```
RUNGE KUTTA DE CUARTO ORDEN
INTERVALO DE INTEGRACION H?0.15
INTRODUCIR X0,Y0,XLAST?0,1
0.15,0.3,0.45,0.6,0.75,0.9,1,
Y(1)=0.71827251

INTERVALO DE INTEGRACION H?.05
INTRODUCIR X0,Y0,XLAST?0,0,1
.05,0.1,0.15,0.2,0.25,0.3,0.35,0
.4,0.45,0.5,0.55,0.6,0.65,0.7,0.
75,0.8,0.85,0.9,0.95,1,1,
Y(1)=0.71828169
```

Discusión de resultados

Se trata de una ecuación diferencial para la que puede encontrarse una solución exacta. La solución es:

Y(1) = e − 2 = 0.718281...

La solución Runge-Kutta es 0,718272... para H = 0,15 y 0,718281 para el intervalo de integración de 0,05. El más corto intervalo de integración proporciona evidentemente una solución más exacta.

OBSERVACIONES SOBRE LA PROGRAMACION

La comprobación de los errores de entrada sobre la magnitud del intervalo se realiza en la línea 60. Obsérvese la visualización del valor final de Y(X) en la línea 245. En la sentencia se utiliza XLAST como su argumento para una mayor claridad. Como se indicó en la descripción del programa, la rutina que comienza en la línea 1000 describe la ecuación diferencial particular objeto de evaluación.

LISTADO DEL PROGRAMA

```
 20 PRINT "RUNGE KUTTA DE CUARTO ORDEN"
 30 PRINT "INTERVALO DE INTEGRACION H?";
 40 INPUT A
 50 PRINT A
 60 IF A<=0 THEN GOTO 30
 70 PRINT "INTRODUCIR XO,YO,XLAST?";
 80 INPUT B
 90 PRINT B;
100 INPUT C
110 PRINT ",";C;
120 INPUT D
130 PRINT ",";D
135 IF B=D THEN GOTO 70
140 LET E=INT ABS (D-B)/A
150 LET F=B
160 LET G=SGN (D-B)*A
170 LET H=C
180 FOR I=1 TO E
190 GOSUB 270
200 NEXT I
210 IF F=D THEN GOTO 240
220 LET G=D-F
230 GOSUB 270
```

```
 240 PRINT
 245 PRINT "Y(";D;")=";H
 250 PRINT
 260 GOTO 30
 270 LET X=F
 280 LET Y=H
 290 GODUB 1000
 300 LET K=G*R
 310 LET X=F+G/2
 320 LET Y=H+K/2
 330 GOSUB 1000
 340 LET L=G*R
 350 LET Y=H+L/2
 360 GOSUB 1000
 370 LET M=G*R
 380 LET X=F+G
 390 LET Y=H+M
 400 GOSUB 1000
 410 LET N=G*R
 420 LET H=H+K/6+L/3+M/3+N/6
 430 LET F=F+G
 440 PRINT F;",";
 450 PAUSE 20
 460 RETURN
1000 LET R=X+Y
1010 RETURN
```

17. Soluciones numéricas de ecuaciones trascendentes

DESCRIPCION DEL PROGRAMA

Este programa resuelve ecuaciones numéricas que, por lo general, no pueden resolverse fácilmente de forma explícita. Por ejemplo, resolverá para X:

$F(X) = LOG\,X - X^2 + 10 = 0.$

El programa solicita una estimación inicial de X0 en las proximidades de la solución. Dicha estimación puede hacerse habitualmente a partir de un trazado de la función. A continuación, solicita un incremento DELTA en el que se evaluará la función (un DELTA grande convergerá más rápidamente, pero puede omitir la solución verdadera). Ahora, el programa solicita EPSILON, un pequeño número positivo. Cuando el valor absoluto de la función es menor que EPSILON, se interrumpe la iteración y se imprime la solución. Si existe más de una solución, el programa encontrará la solución más próxima a la estimación inicial.

El programa utiliza un método de iteración de Newton modificado para encontrar el cruce por cero de la función, que es la solución requerida. El programa comienza por «trazar» una línea recta a través de la función evaluada en X0 + DELTA/2 y X0 – DELTA/2. El cruce de esta línea con el eje X da la siguiente estimación de X0. Si la estimación de X0 interrumpe el cambio mientras el valor absoluto de F(X) siga siendo más grande que EPSILON, el intervalo DELTA se dividirá por la mitad por el programa y continuará la iteración. Es evidente que este método solamente actuará para funciones sin irregularidades entre la primera estimación de X0 y el valor verdadero de X0. La función F(X) ha de ser también de valor único y su primera derivada no puede ser 0 en esta zona; de cualquier otro modo, la línea horizontal recta no cruzará el eje X. Para indicar cómo la iteración sigue adelante (¡con la esperanza de que haya convergencia!), el programa hace parpadear las aproximaciones sucesivas de X0. En tanto que estas últimas sean convergentes, debe llegar a la solución aproximada en un breve período de tiempo. Entonces, el programa se interrumpe y visualiza su mejor estimación de la solución correspondiente a la ecuación.

INSTRUCCIONES

Como en el programa anterior, la ecuación a resolver se añade al programa antes de su ejecución. La subrutina que describe la ecuación debe comenzar en la línea 1000 y ha de finalizar con RETURN, tal como se indica en el listado y ejemplo. Al pulsar la tecla RUN se inicia el programa y genera tres mensajes orientativos, para la conjetura inicial de X0, DELTA y EPSILON. A continuación, la iteración continúa hasta que se verifique ABS(F(X)) < EPSILON.

EJEMPLO

Enunciado del problema

Trataremos de resolver la ecuación antes citada

$$F(X) = \text{LOG}\,X - X^2 + 10 = 0.$$

Utilicemos la conjetura inicial de X0 = 2.0, DELTA = 0.01 y EPSILON = 1E – 6 y luego, se repetirá con una DELTA más grande de 0.1.

Ejemplo de ejecución

```
SOLUCION ECUACION TRANSCENDENTE
INTROD. CONJ. PARA XO,DELTA X,EPS.?
2,.01,1E-6
APROXIMACIONES SUCESIVAS:
3.6656754         3.2673533
3.2421374         3.2420352
3.2420352         3.2420352
3.2420352         3.2420382
3.2420382         3.242039
3.242039          3.2420392

XO=3.2420392


SOLUCION ECUACION TRANSCENDENTE
INTROD. CONJ. PARA XO,DELTA X,EPS.?
2,0.1,1E-6
APROXIMACIONES SUCESIVAS:
3.6650052         3.266929
3.2417394         3.2416374
3.2416374         3.2416374
```

```
3.2419388            3.2419388
3.2419388            3.2420141
3.2420141            3.2420141
3.2420329            3.2420329
3.2420377            3.2420377
3.2420388            3.2420388
3.2420391

X0=3.2420391
```

Discusión del problema

Ambas ejecuciones, después de parpadear los valores convergentes de X, conducen al resultado de X = 3.242...

OBSERVACIONES SOBRE LA PROGRAMACION

El flujo del programa finaliza con STOP en la línea 260 para no interferir con la subrutina que describe la ecuación. Para cada ejecución, se debe pulsar la tecla RUN

LISTADO DEL PROGRAMA

```
 10 PRINT "SOLUCION ECUACION TRANSCENDENTE"
 20 PRINT "INT. CONJ. PARA XO,DELTA X,EPS.?"
 30 INPUT A
 35 LET E=A
 40 PRINT A;
 50 INPUT B
 60 PRINT ",";B;
 70 INPUT C
 80 PRINT ",";C
 85 PRINT "APROXIMACIONES SUCESIVAS:"
 90 LET X=A-B/2
100 GOSUB 1000
110 LET D=Y
120 LET X=A+B/2
130 GOSUB 1000
140 LET Z=B*D/(D-Y)+A-B/2
150 LET X=Z
160 PRINT Z,
170 PAUSE 30
```

```
 180 GOSUB 1000
 190 IF ABS Y<=ABS C THEN GOTO 240
 200 IF Y=E THEN LET B=B/2
 210 LET A=Z
 220 LET E=Y
 230 GOTO 90
 240 PRINT
 245 PRINT
 250 PRINT "XO=";Z
 260 STOP
1000 LET Y=LN X/LN 10-X*X+10
1010 RETURN
```

18. Aritmética de números complejos

DESCRIPCION DEL PROGRAMA

Con este programa, su computadora Timex/Sinclair 1000 se hace una calculadora aritmética de números complejos sofisticada que trabaja en la notación polaca inversa (RPN). Para quienes estén familiarizados con las calculadoras Hewlett Packard, que utilizan RPN, ésta será anticuada. Si no se está familiarizado con RPN se necesitarán algunas explicaciones.

Consideremos una pila de registros, con cuatro en profundidad, denominados X, Y, Z y T en su computadora. La parte inferior de la pila, el registro X y el situado inmediatamente encima, el registro Y, siempre serán objeto de visualización. Cada registro contiene las partes real e imaginaria de un número complejo. Las operaciones usuarias (de una sola variable), como son la elevación al cuadrado, la extracción de raíz cuadrada, el cálculo de recíprocos, etc., se registran solamente en el registro X y no afectan a la pila. Las operaciones con dos variables, tales como las de suma, resta, multiplicación y división, trabajan en los registros X e Y. El resultado de una operación de dos variables aparece siempre en el registro X y descenderá la pila; esto es, el contenido del registro Z se desplaza al registro Y y el registro T se copia en Z. El registro T conserva su valor anterior. Los números se introducen en la pila mediante la orden ENTER (1). Al introducir un número complejo se le desplaza al registro X sustituyendo el contenido anterior de ese registro. Se proveen varias operaciones de manipulación de la pila: X<—>Y intercambia el contenido de los registros X e Y, RAISE STACK destruye el contenido anterior del registro T, desplaza el contenido anterior del registro Z a T, de Y a Z y duplica X en Y. El programa proporciona también un registro de memoria M. Se dispone de dos órdenes para copiar el registro X en M (SAVE) y para copiar el registro M en X (RECALL).

INSTRUCCIONES

RUN — Comenzar el cálculo, emitir llamadas orientativas para las operaciones 1 a 12

1 — ENTER, introduce el número complejo en el registro X y la pila se mantiene invariable

2 — RAISE STACK, copia el contenido de Z en T, de Y en Z y de X en Y; destruye el contenido anterior de T

3 — X<—>Y, intercambia los registros X e Y
4 — SAVE, copia X en M
5 — RECALL, copia M en X
6 — Suma X e Y, pone el resultado en X y hace bajar la pila
7 — Resta X de Y, pone el resultado en X y hace bajar la pila
8 — Multiplica X e Y, pone el resultado en X y hace bajar la pila
9 — Divide Y por X, pone el resultado en X y hace bajar la pila
10 — Calcula el valor recíproco de X y lo pone en X
11 — Calcula el valor de la raíz cuadrada de X y lo pone en X
12 — Calcula el cuadrado de X y lo pone en X

EJEMPLO

Enunciado del problema

Evaluar la expresión siguiente:

Fórmula 18-1

$$\frac{(2 + 3j)\,(5 - 4j)}{5 + 4j}$$

Ejemplo de ejecución

```
OPERACION 1-12?1
INTRODUCIR R,I?2,3
   REAL              IMAGINARIA
Y: 0                 0
X: 2                 3

OPERACION 1-12?3
   REAL              IMAGINARIA
Y: 2                 3
X: 0                 0

OPERACION 1-12?1
INTRODUCIR R,I?5,-4
   REAL              IMAGINARIA
Y: 2                 3
X: 5                 -4

OPERACION 1-12?8
   REAL              IMAGINARIA
Y: 0                 0
X: 22                7
```

```
OPERACION 1-12?3
   REAL             IMAGINARIA
Y: 22               7
X: 0                0

OPERACION 1-12?1
INTRODUCIR R,I?5,4
   REAL             IMAGINARIA
Y: 22               7
X: 5                4

OPERACION 1-12?9
   REAL             IMAGINARIA
Y: 0                0
X: 3.3658537        -1.2926829

OPERACION 1-12?11
   REAL             IMAGINARIA
Y: 0                0
X: 1.8670036        -0.34619187

OPERACION 1-12?12
   REAL             IMAGINARIA
Y: 0                0
X: 3.3658537        -1.2926829
```

Discusión de resultados

Se introduce primero el número complejo 2 + 3j. A continuación se desplaza desde el registro X al Y antes de que se introduzca el segundo número complejo 5-4j. Luego, se multiplian X e Y y el resultado inmediato es 22 + 7j. A continuación el producto se desplaza al registro Y y se introduce el último número complejo 5 + 4j. Después de esta operación, se realiza una división y una raíz cuadrada. El resultado es, entonces: 1.86... – 0.346... j. En el paso final, el resultado se eleva al cuadrado para demostrar que la raíz cuadrada y la elevación al cuadrado de números complejos son operaciones inversas. Obsérvese que existe también una segunda raíz que es la negativa de la primera, a saber:
– 1.86...+ 0.346...j.

OBSERVACIONES SOBRE LA PROGRAMACION

El programa llena la capacidad de RAM de 2K del Timex/Sinclair 1000. Por consiguiente, sólo los registros X e Y se visualizan en lugar de

la pila completa. La función raíz cuadrada (11) visualiza una sola raíz. La otra raíz es la negativa de la primera. La elevación y descenso de la pila se realiza en los lazos FOR NEXT en las líneas 100-110 y 1000-1015. La única comprobación de los errores de entrada se realiza en la línea 27. La bifurcación múltiple se realiza en la línea 40 multiplicando el código de operación por 50.

LISTADO DEL PROGRAMA

```
 10 DIM X(8)
 15 PRINT
 18 PRINT "OPERACION 1-12?";
 20 INPUT F
 25 PRINT F
 27 IF F<>INT F OR F<1 OR F>12 THEN GOTO 18
 40 GOTO 50*F
 50 PRINT "INTRODUCIR R,I?";
 55 INPUT X(1)
 60 PRINT X(1);
 65 INPUT X(2)
 67 PRINT ",";X(2)
 70 LET R=X(1)
 72 LET J=X(2)
 73 PRINT "    REAL","IMAGINARIA","Y:
    ";X(3),X(4),"X: ";R,J
 74 LET D=R*R+J*J
 75 GOTO 15
100 FOR I=8 TO 3 STEP -1
105 LET X(I)=X(I-2)
110 NEXT I
115 GOTO 70
150 LET X(1)=X(3)
165 LET X(2)=X(4)
170 LET X(3)=R
175 LET X(4)=J
180 GOTO 70
200 LET K=R
205 LET M=J
210 GOTO 70
250 LET X(1)=K
255 LET X(2)=M
260 GOTO 70
300 LET X(1)=R+X(3)
305 LET X(2)=J+X(4)
```

```
310 GOTO 1000
350 LET X(1)=X(3)-R
355 LET X(2)=X(4)-J
360 GOTO 1000
400 LET X(1)=R*X(3)-J*X(4)
405 LET X(2)=R*X(4)+J*X(3)
410 GOTO 1000
450 LET X(1)=(R*X(3)+J*X(4))/D
455 LET X(2)=(R*X(4)-J*X(3))/D
460 GOTO 1000
500 LET X(1)=R/D
505 LET X(2)=-J/D
510 GOTO 1000
550 IF R<>0 OR J<>0 THEN GOTO 554
551 LET X(1)=0
552 LET X(2)=0
553 GOTO 70
554 LET X(1)=SQR ((R+SQR (R*R+J*J))/2)
555 LET X(2)=J/2/X(1)
560 GOTO 70
600 LET X(1)=R*R-J*J
605 LET X(2)=2*R*J
610 GOTO 70
1000 FOR I=3 TO 6
1010 LET X(I)=X(I+2)
1015 NEXT I
1020 GOTO 70
```

19. Paso de fracciones decimales a fracciones binarias

DESCRIPCION DEL PROGRAMA

La parte del mundo civilizado que emplea el sistema británico de unidades de media todavía prefiere las fracciones binarias tales como 5/16 ó 7/32 a los números tan fáciles de recordar como 0.3125 ó 0.21875. Para atender a ambos sistemas, este programa convertirá una fracción decimal en su equivalente en fracción binaria. Definiremos una «fracción binaria» como una fracción cuyo denominador es una potencia entera de 2. Todo lo que el usuario del programa tiene que hacer es indicar la más pequeña fracción aceptable con un denominador potencia binaria de 2, v.g. 256 por 1/256, 64 por 1/64, etc.

Si la conversión puede hacerse con exactitud, el programa dividirá el numerador por el denominador. Por ejemplo, 0.375 se convertirá a 3/8, aunque la más pequeña fracción requerida por el usuario puede ser 1/256. Si la conversión no puede realizarse con exactitud, el más pequeño denominador binario se encontrará dentro de las limitaciones dadas y se indicará el error porcentual.

INSTRUCCIONES

Al pulsar la tecla RUN se emiten dos llamadas de mensaje orientativo: INTRODUCIR FRACCION DECIMAL? (el número a convertir en la fracción binaria y DENOMINADOR MAX. (POT. DE 2) (el número a indicar el mayor denominador admisible). Entonces, el programa visualiza la fracción binaria y el error, si lo hubiere.

EJEMPLO

Enunciado del problema

Convertir 2:5, 0.12717 y 0.1875 en fracciones binarias. No aceptar ninguna fracción más pequeña que 1/16 para el primer ejemplo y 1/256 para los ejemplos segundo y tercero.

Ejemplo de ejecución

```
INTRODUCIR FRACCION DECIMAL?2.5
DENOMINADOR MAX (POT. DE 2)?16
2.5=5/2

INTRODUCIR FRACCION DECIMAL?0.12717
DENOMINADOR MAX. (POT. DE 2)?256
0.12717 APROXIMA 33/256
CON ERROR DE -1.3652984 POR CIENTO

INTRODUCIR FRACCION DECIMAL?0.1875
DENOMINADOR MAX. (POT. DE 2)?256
0.1875=3/16
```

Discusión de resultados

Los ejemplos primero y tercero conducen a fracciones binarias exactas. El segundo ejemplo puede aproximarse dentro de las limitaciones de 1/256 para dentro de 1.35...%.

OBSERVACIONES SOBRE LA PROGRAMACION

La única comprobación de errores con respecto al valor de entrada (denominador más grande) se realiza en las líneas 70-90. Para aplicar el redondeo del Timex/Sinclair 1000 se utiliza un factor «ficticio» de 0.0001 en las líneas 90 y 120. Para encontrar el más pequeño denominador, el programa hace que se divida el numerador y el denominador por 2 (líneas 140-150), en tanto que ambos sean enteros.

LISTADO DEL PROGRAMA

```
  5 PRINT
 10 PRINT "INTRODUCIR FRACCION DEC.?";
 20 INPUT A
 30 PRINT A
 40 PRINT "DENOMINADOR MAX. (POT. DE 2)?";
 50 INPUT B
 60 PRINT B
 70 IF B<2 THEN GOTO 40
 80 LET C=LN B/LN 2
 90 IF ABS (C-INT C)>.0001 THEN GOTO 40
100 LET C=INT C
```

```
110 LET D=A*B
120 IF (ABS D-INT ABS (D+.0001)>=.0001)
    THEN GOTO 170
130 IF D/2<>INT (D/2) THEN GOTO 230
140 LET D=D/2
150 LET B=B/2
160 GOTO 130
170 LET D=INT (D+.5)
180 LET E=A-D/B
190 LET F=100*E/A
200 PRINT A;" APROXIMA ";D;"/";B
210 PRINT "CON ERROR DE ";F;" POR CIENTO"
220 GOTO 5
230 PRINT A;"=";D;"/";B
240 GOTO 5
```

INVESTIGACION OPERATIVA

20. Solución de ecuaciones de las colas

DESCRIPCION DEL PROGRAMA

Este programa resuelve las ecuaciones de las colas M/M/S que describen muchas situaciones de la vida real.

Estas ecuaciones suponen una distribución exponencial de las tasas de llegada de clientes L, una distribución exponencial de los tiempos de servir M, S sirvientes idénticos y un orden de servicio FIFO (primero en entrar, primero en salir). La distribución exponencial implica independencia de eventos; los clientes buscarán servicio con independencia de la longitud de la cola y los sirvientes trabajarán a un régimen constante haciendo caso omiso de la carga y de la longitud de la cola. Dichos supuestos suelen proporcionar estimaciones seguras (pesimistas) de los tiempos de espera y de otros parámetros de colas. Los casos de M/M/S dan lugar a ecuaciones relativamente simples (en términos de la teoría de las colas) y pueden resolverse dentro de las limitaciones de memoria del Timex/Sinclair 1000. El programa puede utilizarse para estimar problemas tales como cuántos sillones de barbero hay que proporcionar para un establecimiento situado en el centro de la ciudad, cuántas operadoras de teléfonos deben trabajar entre las 2 y las 3 de la tarde o cuántas impresoras deben conectarse a un procesador central. Por lo general, la aplicación del programa ayudará a compensar el coste de proporcionar servicio con la incomodidad de largas colas de espera. Las ecuaciones de las colas utilizadas en el programa se dan a continuación.

Limitaciones del programa: $S < 34$, $U <= 1.0$, todas las entradas positivas, todos los resultados intermedios deben ser inferiores a 1.0E38.

Entradas:
Tasa de llegada media *r*
Tasa de servicio media *m*
Número de sirvientes *s*
Números de elementos en sistemas *n*

Salidas:
Utilización de las instalaciones U = r/(ms)
Prob. de no encontrar ningún elemento en sistema P(0)

$$P(0) = ((r/m)^s /s!(1\text{-}U) + \sum_{j=0}^{s-1} (r/m)^j /j!)^{-1}$$

Prob. de encontrar *n* elementos en el sistema P(n)

$$P(n) = P(0) \times (r/m)^n \times (1/n!) \quad \text{para} \quad n<s$$
$$P(n) = P(0) \times (r/m)^n \times (1/(s!s^{n-s})) \quad \text{para} \quad n>=s$$

Prob. de encontrar ocupados a todos los sirvientes B

$$B = \sum_{n=s}^{\infty} P(n) = (r/m)^s \times P(0)/(s! \times (1\text{-}U))$$

Tiempo de espera medio en cola Tw = B/(sm × (1 – U))
Tiempo de respuesta medio del sistema Tr = Tw + 1/m
Dev. estánd. de Tr = Tw/B × $\sqrt{(B(2 - B) + s^2 \times (1 - U)^2)}$
Prob. de Tw > T = B × Exp (– smT × (1 – U))
Prob. de Tr > T = Exp (– mT) × (1 + (1 – Exp (– msTk)) × B/sk) para k < > 0 en donde k = 1 – U – I/s
Prob. de Tr > T = Exp (– mT) × (1 + BmT) para k = 0
Número medio de elementos en cola Q = r × Tw
Número medio de elementos en sistema N = r × Tr

El programa proporciona todas las llamadas de mensajes orientativos y comprueba los datos de entrada para ver si los valores son correctos.

INSTRUCCIONES DEL PROGRAMA

Al pulsar la tecla RUN emitirá llamadas orientativas para introducir la tasa de llegada L, la tasa de servicio M y el número de servicio S. Una entrada incorrecta hará que se repitan las llamadas orientativas. A continuación, el programa calcula y visualiza los parámetros anteriormente citados del sistema de las colas, a saber B, U, P(0), Tw, Tr, desviación standard de Tr, Q y N. Entonces, el programa emitirá un mensaje orientativo: «INTRODUCIR COD 1, 2 OR 3?». Para calcular P(n), introducir 1,P(Tw, Tr)– 2 o para continuar con una nueva ejecución – 3. Al seleccionar P(n) se generará un mensaje orientativo para el número de elementos en el sistema n y calculará y visualizará P(n). Al elegir P(Tw,Tr) se emitirá un mensaje para el tiempo T y calculará y visualizará P(Tw > T) y P(Tr > T). A continuación, se repiten los mensajes orientativos.

EJEMPLO

Enunciado del problema

Una zona de trabajo de computadora tiene cinco terminales. Por término medio, 16 clientes por hora llegan a intervalos aleatorios con cada

trabajo teniendo una desviación media de 15 minutos (cuatro por hora). Calcular los parámetros de las colas y encontrar también la reducción en el tiempo de espera si se instala un sexto terminal.

Ejemplo de ejecución

```
INTRODUCIR L,M,S?16,4,5
P(TODO OCUPADO)=      0.55411255
UTIL.=                0.8
P(0)=                 .012987013
ESPERA MED.=          0.13852814
RESP.MED.=            0.38852814
DESV.ESTAND.(TR)=     0.33552053
MED. EN SIST.=        6.2164502
MED. EN COLA=         2.2164502

INTRODUCIR CODIGO 1,2 O 3?1
NO. ELEM. EN SISTEMA?1
P(N=1)=.051948052

INTRODUCIR CODIGO 1,2 O 3?1
NO. ELEM. EN SISTEMA?6
P(N=6)=.08865809

INTRODUCIR CODIGO 1,2 O 3?1
NO. ELEM. EN SISTEMA?10
P(N=10)=.03631432

INTRODUCIR CODIGO 1,2 O 3?2
TIEMPO=?0.2
P(TW>0.2)=0.24897882
P(TR>0.2)=0.64851202

INTRODUCIR CODIGO 1,2 O 3?2
TIEMPO=?10
P(TW>10)=2.3540663E-18
P(TR>10)=9.8411012E-17

INTRODUCIR L,M,S?16,4,6
P(TODO OCUPADO)= 0.28476085
UTIL.=           0.66666667
P(0)=            .016685206
ESPERA MED.=     .035595106
RESP.MED.=       0.28559511
DEV.ST.(TR)=     0.26482403
```

```
MED. EN SIST.=    4.5695217
MED. EN COLA=     0.56952169

INTRODUCIR COD. 1,2 O 3?1
NO. ELEMENTOS EN SIST.?1
P(N=1)=.066740823

INTRODUCIR CODIGO 1,2 O 3?2
TIEMPO=?0.2
P(TW>0.2)=.057492223
P(TR>0.2)=0.51978804
```

Discusión de resultados

La primera parte del ejemplo (cinco terminales) muestra una utilización bastante alta de las instalaciones, 0.8. El tiempo de espera medio en cola es 0.318 o algo más de 8 minutos. La adición de un sexto terminal producirá una mejora considerable. El tiempo de espera disminuirá a 0.035 horas o unos dos minutos. Todas las demás probabilidades de espera resultarán también afectadas de forma correspondiente.

OBSERVACIONES SOBRE LA PROGRAMACION

Una comprobación de los valores de entrada se realiza en la línea 80. Obsérvese el empleo del factor ficticio de redondeo en la línea 60. Una sentencia PRINT bastante complicada se muestra en la línea 900. A medida que se introduzca el programa se observará el empleo de la función EXP (desplazar ENTER para introducir el modo FUNCION, y luego pulsar la tecla X).

LISTADO DEL PROGRAMA

```
 5 CLS
10 PRINT "INTRODUCIR L,M,S?";
20 INPUT L
30 PRINT L;
40 INPUT M
50 PRINT ",";M;
60 INPUT S
70 PRINT ",";S
80 IF L<=0 OR M<=0 OR S<=0 OR S<>INT S OR S>33
   THEN GOTO 10
85 LET P=0
```

```
 90 LET X=L/M
100 LET U=X/S
110 LET V=1-U
120 IF V<=0 THEN GOTO 10
130 LET Y=S
140 GOSUB 1000
150 LET T=Z
160 FOR I=0 TO S-1
170 LET Y=I
180 GOSUB 1000
190 LET P=P+X**I/Z
200 NEXT I
210 LET Y=S
220 GOSUB 1000
230 LET P=1/(P+X**S/T/V)
240 LET B=X**S*P/T/V
250 LET W=B/S/M/V
260 LET R=W+1/M
270 PRINT "P(TODO OCUPADO)=",B,"UTIL=",U,
    "P(0)=",
275 PRINT P,"ESPERA MED.=",W,"RESP.MED.=",R
280 PRINT "DESV.ESTAND.(TR)=",SQR(2*B-2*B+(S*V)
    **2)*W/B
290 PRINT "MED. EN SIST.=",L*R,"MED.
    EN COLA=",L*W
350 PRINT
355 PRINT "INTRODUCIR CODIGO 1,2 O 3?";
360 INPUT K
370 PRINT K
380 IF K<>1 THEN GOTO 510
390 PRINT "NO. ELEM. EN SISTEMA?";
400 INPUT N
410 PRINT N
420 LET D=P*X**N
430 IF N<S THEN GOTO 460
440 LET E=D/T/S**(N-S)
450 GOTO 490
460 LET Y=N
470 GOSUB 1000
480 LET E=D/Z
490 PRINT "P(N=";N;")=";E
500 GOTO 350
510 IF K<>2 THEN GOTO 5
520 PRINT "TIEMPO=?";
530 INPUT H
```

```
 540 PRINT H
 550 LET F=B*EXP -(S*M*V*H)
 560 LET A$="W"
 570 GOSUB 900
 580 LET A$="R"
 590 LET G=V-1/S
 600 LET D=EXP -(M*H)
 610 IF ABS G<=.00001 THEN GOTO 640
 620 LET F=D*(1+B/S*(1-EXP -(M*S*G*H))/G)
 630 GOTO 650
 640 LET F=D*(1+B*M*H)
 650 GOSUB 900
 660 GOTO 350
 900 PRINT "P(T";A$;">";H;")=";F
 910 RETURN
1000 LET Z=1
1010 FOR J=1 TO Y
1020 LET Z=Z*J
1030 NEXT J
1040 RETURN
```

21. Parámetros de fiabilidad de sistemas complejos

DESCRIPCION DEL PROGRAMA

La fiabilidad de un sistema compuesto (v. g. central telefónica constituida por enlaces interurbanos de entrada y de salida, dispositivos de conmutación y fuentes de alimentación o un centro de cálculo constituido por unidades CPU, impresoras, unidades de disco y fuentes de alimentación) puede expresarse en función del Tiempo medio antes del fallo (MTBF) y del Tiempo medio hasta la reparación (MTTR). Un alto MTBF y un bajo MTTR puede conseguirse utilizando componentes fiables que no se deterioren con frecuencia y sean fáciles de reparar o proporcionando redundancia. Por ejemplo, dos unidades centrales de proceso (CPU) pueden compartir la carga; si una falla, la otra ocupará su lugar.

El programa determinará la fiabilidad total de un sistema dados los parámetros MTBF y MTTR de componentes individuales. Se supone que los fallos de componentes individuales son independientes entre sí. La curva de fiabilidad sigue, entonces, la parte central de la curva de «pila de baño» típica, en donde el alto número de fallos al principio es causado por problemas iniciales y el alto número de fallos hacia el final de la vida útil es causado por la degradación por el uso del componente. También se supone que los componentes individuales se agruparán en subsistemas de uno de los tres tipos citados a continuación y que varios subsistemas están conectados en serie para formar un sistema total.

A. Subsistemas en serie. El fallo de un solo componente incapacitará el sistema completo.

B. Subsistemas en paralelo. Al menos un componente ha de mantenerse activo para que el sistema pueda funcionar.

C. Subsistema K de n en paralelo. En donde K componentes de entre n han de estar en buen estado (activos) para que el sistema funcione.

El programa calcula MTBF y MTTR para cada subsistema y para el sistema total. También determina el factor de disponibilidad y la probabilidad de que x componentes fallen durante el tiempo t, en donde pueden seleccionarse x y t.

Se utilizan las fórmulas siguientes:

A–Disponibilidad
U–No disponibilidad
N–n.º de elementos en subsistema
k–n.º de elementos requeridos para la explotación del sistema
ñ–n.º de fallos durante el tiempo t (valor medio)
P(k,t)–Prob. de k fallos durante el tiempo t
i–elemento, s–subsistema, subíndice

Fórmulas generales:

$$A = \frac{MTBF}{MTBF + MTTR} \qquad \bar{n} = t/MTBF$$

$$U = 1 - A \qquad P(k,t) = \frac{e^{-\bar{n}} (\bar{n})^k}{k!}$$

Subsistemas en serie:

$$A_s = \prod_1^N A_i \qquad MTBF_s = 1/\sum_1^N (1/MTBF_i) \qquad MTTR_s = \frac{1 - A_s}{A_s} MTBF_s$$

Subsistemas en paralelo:

$$A_s = 1 - \prod_1^N (1 - A_i) \qquad MTBF_s = \frac{A_s}{1 - A_s} MTTR_s$$

$$MTTR_s = 1/\sum_1^N (1/MTTR_i)$$

«K de N» en paralelo

$$MTTR_s = MTTR_i/(N-K+1)$$

$$MTBF_s = MTBF_i \left(\frac{MTBF_i}{MTTR_i}\right)^{(N-K)} \times \frac{(N-K)!\ (K-1)!}{N!}$$

INSTRUCCIONES

Pulsar la tecla RUN para iniciar el programa y luego, seguir el mensaje orientativo introduciendo un número 0 y 6.

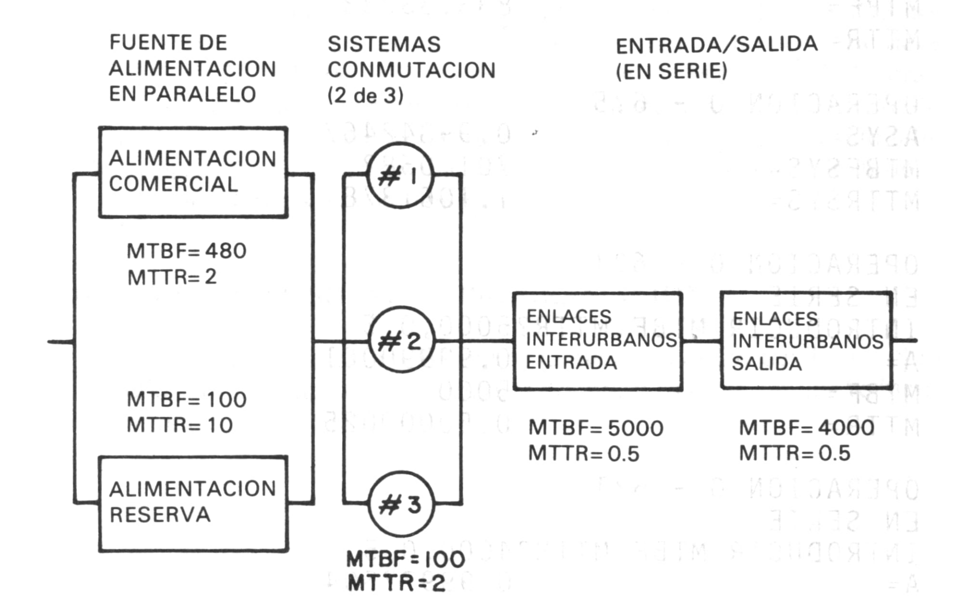
0 — Comenzar una nueva ejecución
1 — Introducir MTBF y MTTR para un subsistema en serie
2 — Introducir MTBF y MTTR para un subsistema en paralelo redundante
3 — Introducir MTBF y MTTR para un subsistema k de n en paralelo redundante
4 — Borrar subsistema si se cometió un error durante la introdución
5 — Incluir un subsistema en un sistema total
6 — Calcular la probabilidad de x fallos durante el tiempo t.

EJEMPLO

Enunciado del problema

Considérese una instalación de central telefónica, tal como se muestra en la figura 21-1, que está constituida por una fuente de alimentación y otra de reserva, tres sistemas de conmutación, dos de los cuales se requieren para el funcionamiento, un grupo de enlaces interurbanos de entrada y otro de salida conectados a antenas de microondas, todo con parámetros MTBF y MTTR conocidos. Calcular los parámetros de fiabilidad total incluyendo la probabilidad de fallo en 12 y en 24 horas.

Figura 21-1



Ejemplo de ejecución

```
OPERACION 0 - 6?2
EN PARALELO
INTRODUCIR MTBF,MTTR?480,2
A=                    0.99585062
MTBF=                 480
MTTR=                 2

OPERACION 0 - 6?2
EN PARALELO
INTRODUCIR MTBF,MTTR?100,10
A=                    0.99962278
MTBF=                 4416.6667
MTTR=                 1.6666667

OPERACION 0 - 6?5
ASYS=                 0.99962278
MTBFSYS=              4416.6667
MTTRSYS=              1.6666677
OPERACION 0 - 6?3
K-OF-N
INTRODUCIR MTBF,MTTR?100,2
INTRODUCIR K,N?2,3
A=                    0.99880144
MTBF=                 833.33333
MTTR=                 1

OPERACION 0 - 6?5
ASYS=                 0.99842467
MTBFSYS=              701.0582
MTTRSYS=              1.1061378

OPERACION 0 - 6?1
EN SERIE
INTRODUCIR MTBF,MTTR?5000,0.5
A=                    0.99990001
MTBF=                 5000
MTTR=                 0.50000025

OPERACION 0 - 6?1
EN SERIE
INTRODUCIR MTBF,MTTR?4000,0.5
A=                    0.99977504
```

```
A=                    0.99977504
MTBF=                 2222.2222
MTTR=                 0.50002885

OPERACION 0 - 6?5
ASYS=                 0.99820007
MTBFSYS=              532.93112
MTTRSYS=              0.96097057

OPERACION 0 - 6?6
TIEMPO,NO.FALLOS?24,1
P(24,1)=.043050892

OPERACION 0 - 6?6
TIEMPO.NO.FALLOS?12,1
P(12,1)=.022015632

OPERACION 0 - 6?6
TIEMPO,NO.FALLOS?24,2
P(24,2)=.00096937613
```

Discusión de resultados

Al examinar el subsistema, vemos primero que el subsistema de fuentes de alimentación en paralelo tiene un MTBF de 4416 horas, el subsistema de dispositivos de conmutación de 2 de entre 3 tiene un MTBF de 833 horas y el subsistema de enlaces interurbanos de entrada/salida en serie tiene un MTBF de 2222 horas. El sistema total tiene un MTBF de 532 horas. Para mejorar este factor se requeriría evidentemente una mejora en la fiabilidad de los dispositivos de conmutación. La probabilidad de un solo fallo en 24 horas es 4,3%, de un solo fallo en 12 horas es 2,2%. La probabilidad de 2 fallos en 24 horas es solamente de 0.097%.

OBSERVACIONES SOBRE LA PROGRAMACION

El programa llena casi la totalidad de la memoria de 2K del Timex/Sinclair 1000: Por este motivo, los mensajes orientativos tuvieron que reducirse al mínimo.

LISTADO DEL PROGRAMA

```
 15 GOSUB 800
 20 LET M=1
 40 LET N=0
 50 PRINT
 60 PRINT "OPERACION 0 - 6?";
 70 INPUT Z
 80 PRINT Z
 90 GOTO 100*Z+10
110 PRINT "EN SERIE"
120 GOSUB 1000
130 LET C=C+1/A
140 LET D=D*A/(A+B)
150 LET F=1/C
160 LET H=F*(1-D)/D
170 LET E=D
180 PRINT "A=",E,"MTBF=",F,"MTTR=",H
190 GOTO 50
210 PRINT "EN PARALELO"
215 GOSUB 1000
220 LET C=C+1/B
225 LET D=D*B/(A+B)
230 LET E=1-D
240 LET H=1/C
250 LET F=E*H/D
260 GOTO 180
310 PRINT "K-OF-N"
315 GOSUB 1000
320 PRINT "INTRODUCIR K,N?";
325 INPUT K
330 PRINT K;
335 INPUT R
340 PRINT ",";R
345 LET L=R-K
350 GOSUB 900
355 LET F=J
360 LET L=K-1
365 GOSUB 900
370 LET F=F*J
375 LET L=R
380 GOSUB 900
385 LET F=A*F/J*(A/B)**(R-K)
390 LET H=B/(R-K+1)
395 LET E=F/(F+H)
```

```
400 GOTO 180
410 GOSUB 800
420 PRINT "SUBSISTEMA BORRADO"
430 GOTO 50
510 GOSUB 800
520 LET M=M*E
530 LET N=N+1/F
540 LET P=(1-M)/M/N
550 LET X=1/N
560 PRINT "ASYS=",M,"MTBFSYS=",X,"MTTRSYS=",P
570 GOTO 50
610 PRINT "TIEMPO,NO.FALLOS?";
612 INPUT T
614 PRINT T;
615 INPUT Q
620 PRINT ",";Q
625 LET S=T*N
630 LET L=Q
635 GOSUB 900
640 LET U=EXP -S*S**Q/J
645 PRINT "P(";T;",";Q;")=";U
650 GOTO 50
800 LET C=0
810 LET D=1
820 RETURN
900 LET J=1
910 FOR I=1 TO L
920 LET J=J*I
930 NEXT I
940 RETURN
1000 PRINT "INTRODUCIR MTBF,MTTR?";
1010 INPUT A
1020 PRINT A;
1030 INPUT B
1040 PRINT ",";B
1050 RETURN
```

MISCELANEA

22. Frecuencia de las notas musicales en la escala bien templada

DESCRIPCION DEL PROGRAMA

Este programa determina la frecuencia de una nota musical dada en la escala bien afinada. La escala supone una relación de frecuencias constante de la raíz 12-ésima de 2 entre dos notas contiguas. Johann Sebastián Bach estaba tan excitado con la idea de la escala bien templada (afinada) que escribió una serie de composiciones de piano en la clave de cada nota en esa escala. Las octavas en el programa están numeradas como enteros negativos o positivos, o 0, con la nota A («la») en la octava 0 con la frecuencia de 440 Hz. La frecuencia de las notas se dobla con cada octava sucesivamente más alta.

INSTRUCCIONES

Al pulsar la tecla RUN se inicia el programa y se emiten dos mensajes orientativos correspondientes a la nota y al número de la octava. La nota debe introducirse como C, C +, D, D +, E, F, F +,G, G +,A, A + o B; la octava como – 2, – 1,0, 1,2... etc. Como el Timex/Sinclair 1000 no tiene una tecla de sostenido (≠), este signo se sustituye por (+).

EJEMPLO

Enunciado del problema

Determinar las frecuencias de la nota A («la») en la octava 0, C+ en la tercera octava, D en la octava – 4 y C y B en la primera octava (1).

---

(1) Nota del T.: Las equivalencias de la nomenclatura anglosajona de las notas musicales es la siguiente:

C-«do»; D-«re»; E-«mi»; F-«fa»; G-«sol»; A-«la», B-«si»; C+ =«do» sostenido; D+ = «re» sostenido; F+ = «fa» sostenido; G+ = «sol» sostenido; A+ = «la» sostenido.

Ejemplo de ejecución

```
INTRODUCIR NOTA E.G. C,C+,D..?A
INTRODUCIR OCTAVA?0
A,OCTAVA 0=440 HZ

INTRODUCIR NOTA E.G, C,C+,D..?C+
INTRODUCIR OCTAVA?3
C+,OCTAVA 3=2217.4611 HZ

INTRODUCIR NOTA E.G. C,C+,D..?D
INTRODUCIR OCTAVA?-4
D,OCTAVA -4=18.354048 HZ

INTRODUCIR NOTA E.G. C,C+,D..?K
INTRODUCIR OCTAVA?3
NOTA ERRONEA

INTRODUCIR NOTA E.G. C,C+,D..?C
INTRODUCIR OCTAVA?1
C,OCTAVA 1=523.25113 HZ

INTRODUCIR NOTA E.G. C,C+,D..?B
INTRODUCIR OCTAVA?1
B,OCTAVA 1=987.7666 HZ
```

Discusión de resultados

El primer ejemplo, como se esperaba, proporciona 440 Hz, la cuarta entrada muestra una introdución errónea (K) que proporciona un retorno de error y la repetición del mensaje orientativo.

OBSERVACIONES SOBRE LA PROGRAMACION

En este caso, para cambiar, se trata de un programa corto. Las líneas 100 y 135 realizan la comprobación de errores de las entradas. Si la designación de las notas no se reconociera como legítima, entonces, la variable I mantiene su valor asignado de 100 y se visualiza la observación (comentario) «NOTA ERRONEA». El cálculo de la frecuencia se realiza en la línea 170.

LISTADO DEL PROGRAMA

```
 10 CLS
 50 PRINT
 52 PRINT "INTRODUCIR NOTA E.G. C,C+,D..?";
 60 INPUT B$
 70 PRINT B$
 80 PRINT "INTRODUCIR OCTAVA?";
 90 INPUT B
100 IF B<>INT B THEN GOTO 80
110 PRINT B
120 LET I=100
121 IF B$="C" THEN LET I=1
122 IF B$="C+" THEN LET I=2
123 IF B$="D" THEN LET I=3
124 IF B$="D+" THEN LET I=4
125 IF B$="E" THEN LET I=5
126 IF B$="F" THEN LET I=6
127 IF B$="F+" THEN LET I=7
128 IF B$="G" THEN LET I=8
129 IF B$="G+" THEN LET I=9
130 IF B$="A" THEN LET I=10
131 IF B$="A+" THEN LET I=11
132 IF B$="B" THEN LET I=12
135 IF I<>100 THEN GOTO 170
150 PRINT "NOTA ERRONEA"
160 GOTO 50
170 LET F=2**((I-10)/12+B)*440
180 PRINT B$;",OCTAVA ";B;"=";F;" HZ"
190 GOTO 50
```

23. Juego de estrategia de inteligencia artificial

DESCRIPCION DEL PROGRAMA

Este programa, que es una versión del popular juego de CARA o CRUZ es bastante singular por cuanto que aprende jugando con el oponente humano. De este modo, demuestra un fuerte rasgo de inteligencia.

El juego consiste en que el jugador elige CARA o CRUZ. Si la computadora tiene una respuesta correcta al hacer la misma elección, entonces ganará la computadora. Si la elección es diferente, ganará el jugador. Si usted, el jugador, juega al azar, la computadora también lo hará en sus intentos de acierto y el número de veces en ganar o perder será casi el mismo. Sin embargo, si sigue alguna de varias estrategias simples; v.g. jugar siempre CARA o jugar siempre CRUZ hasta que pierda, y entonces cambiar a CARA y viceversa, o jugar lo mismo (CARA o CRUZ) si pierde, pero de forma alternada si gana, la computadora adivinará su estrategia después de unas pocas jugadas y le derrotará todas las veces desde entonces, hasta que cambie su estrategia. Tan pronto como cambie su estrategia y pierda la computadora, esta última comenzará a jugar al azar hasta que «adivine» su nueva estrategia y le vuelva a ganar.

El programa comprueba las dos jugadas anteriores en el juego y consulta su memoria para el mismo tipo de situación anteriormente jugada. Si se encuentra una situación anterior, la computadora hará un intento de acierto basándose en la experiencia anterior. Si entonces, el jugador actúa según lo previsto, ganará la computadora. Si la situación no se produjo con anterioridad, la computadora jugará al azar y almacenará el resultado de su jugada con la respuesta del jugador para establecer una nueva situación.

INSTRUCCIONES

Al pulsar la tecla RUN se emite un mensaje orientativo de sugerencia para quien inicializa la función RND del BASIC. El valor de inicialización debe estar comprendido entre 0 y 65535. La introducción de 0 comenzará siempre una secuencia aleatoria diferente; de cualquier otro modo, la secuencia estará previamente determinada. A continuación, el programa emite la llamada indicativa de H (cara) o T (cruz). Finalmente, anuncia quién ganó, muestra la puntuación total y continúa con la siguiente jugada.

EJEMPLO

Enunciado del problema

El jugador (usted) seguirá primero la estrategia de alternar cuando pierda y de jugar lo mismo cuando gane. A continuación, cuando la computadora adivine su estrategia, invertirá esta última.

Ejemplo de ejecución

```
INICIADOR?1
(H)CARA O (T)CRUZ?H
JUEGO 1,TU GANAS, ACIERTOS=1
(H)CARA O T(CRUZ)?H
JUEGO 2,TU GANAS, ACIERTOS=2
(H)CARA O (T)CRUZ?H
JUEGO 3,TU PIERDES, ACIERTOS=1
(H)CARA O (T)CRUZ?T
JUEGO 4,TU GANAS, ACIERTOS=2
(H)CARA O (T)CRUZ?T
JUEGO 5,TU GANAS, ACIERTOS=3
(H)CARA O (T)CRUZ?T
JUEGO 6,TU PIERDES, ACIERTOS=2
(H)CARA O (T)CRUZ?H
JUEGO 7,TU PIERDES, ACIERTOS=1
(H)CARA O (T)CRUZ?T
JUEGO 8,TU PIERDES, ACIERTOS=0
(H) CARA O (T)CRUZ?H
JUEGO 9,TU PIERDES, ACIERTOS=-1
(H)CARA O (T)CRUZ?T
JUEGO 10,TU PIERDES, ACIERTOS=-2
(H)CARA O (T)CRUZ?T
JUEGO 11,TU GANAS, ACIERTOS=-1
(H)CARA O (T)CRUZ?H
JUEGO 12,TU GANAS, ACIERTOS=0
(H)CARA O (T)CRUZ?T
JUEGO 13,TU PIERDES, ACIERTOS=-1
(H)CARA O (T)CRUZ?T
JUEGO 14,TU PIERDES, ACIERTOS=-2
(H)CARA O (T)CRUZ?T
JUEGO 15,TU PIERDES, ACIERTOS=-3
(H)CARA O (T)CRUZ?T
JUEGO 16,TU PIERDES, ACIERTOS=-4
(H)CARA O (T)CRUZ?T
JUEGO 17,TU PIERDES, ACIERTOS=-5
```

```
JUEGO 18,TU GANAS, ACIERTOS=-4
(H)CARA O (T)CRUZ?H
JUEGO 19,TU PIERDES, ACIERTOS=-5
(H)CARA O (T)CRUZ?T
JUEGO 20,TU PIERDES, ACIERTOS=-6
(H)CARA O (T)CRUZ?H
JUEGO 21,TU GANAS, ACIERTOS=-5
(H)CARA O (T)CRUZ?H
JUEGO 22,TU PIERDES, ACIERTOS=-6
(H)CARA O (T)CRUZ?T
JUEGO 23,TU PIERDES, ACIERTOS=-7
(H)CARA O (T)CRUZ?H
JUEGO 24,TU PIERDES, ACIERTOS=-8
(H)CARA O (T)CRUZ?
```

Discusión de resultados

Sigue su primera estrategia hasta el juego 10. La computadora está ganando cada juego, comenzando con el juego 6. Cuando cambia de estrategia en el juego 11, ganará usted primero pero la computadora vuelve a adivinar su estrategia comenzando con el juego 13. Usted volverá a su antigua estrategia en el juego 18. Después de unas pocas jugadas, la computadora vuelve a descubrir su estrategia y después de 24 juegos usted llegará a tener un conteo negativo de 8 puntos.

OBSERVACIONES SOBRE LA PROGRAMACION

Los resultados de diversos juegos se almacenarán en la variable dimensionada A. La comprobación de la entrada correcta (H) o (T) se realiza en la línea 200. El programa requiere el modo SLOW (lento) de modo que los resultados puedan visualizarse continuamente durante la secuencia de INKEY$.

LISTADO DEL PROGRAMA

```
10 CLS
15 DIM A(8)
18 SLOW
20 PRINT "INICIADOR?";
30 INPUT X
40 PRINT X
45 RAND (X)
50 LET J=0
```

```
 60 LET I=0
 70 LET I=I+1
 80 IF I<3 THEN GOTO 140
 90 LET L=INT ((M*N+1)/2)
100 LET W=4*Q+2*L+P+1
110 IF A(W)=0 THEN GOTO 140
120 LET S=M*A(W)
130 GOTO 170
140 LET S=2*INT (2*RND)-1
170 PRINT "(H)CARA O (T)CRUZ?";
180 IF INKEY$="" THEN GOTO 180
185 LET T$=INKEY$
190 IF INKEY$=T$ THEN GOTO 190
200 IF T$<>"T" AND T$<>"H" THEN GOTO 180
205 PRINT T$
210 LET U=1
220 IF T$="T" THEN LET U=-1
230 LET J=J-S*U
240 LET Z=2-I
250 IF Z=0 THEN GOTO 320
260 IF Z>0 THEN GOTO 340
270 IF A(W)=0 THEN GOTO 310
280 IF S=U THEN GOTO 320
290 LET A(W)=0
300 GOTO 320
310 LET A(W)=M*U
320 LET N=M
330 LET Q=P
340 LET M=U
350 LET P=INT ((S*U+1)/2)
360 LET A$="GANAS,"
370 IF S=U THEN LET A$="PIERDES,"
380 PRINT "JUEGO ";I;",TU ";A$;"=";J
390 GOTO 70
```

24. Tiradas de dados múltiples

DESCRIPCION DEL PROGRAMA

Muchos juegos (v. g. el popular YAHTZEE, e incluso simulaciones de modelos de computadoras están basados en el resultado de una tirada de datos múltiples. Este programa permite una tirada de varios dados a la vez y visualiza el resultado de cada tirada múltiple: El generador pseudo-aleatorio RND del BASIC del Timex/Sinclair 1000 en la línea 90, que hace la «tirada», trabaja bien en esta aplicación; sin embargo, si tiene su propio favorito, no vacile en sustituir esa línea.

INSTRUCCIONES

Al pulsar al tecla RUN se visualiza CUANTOS DADOS para la tirada correspondiente. A continuación se introduce un número n (uno o más), el programa visualiza n dígitos, en donde cada dígito está comprendido entre 1 y 6 y corresponde a los puntos en un dado. A continuación, aparecerá en la pantalla la visualización antes indicada, CUANTOS DADOS, para otra tirada.

EJEMPLO

Enunciado del problema

Mantener la tirada de dados, tal como se indica en el listado.

Ejemplo de ejecución

```
DADOS MULTIPLES

CUANTOS DATOS?5
4,2,2,3,5
CUANTOS DADOS?3
6,6,1
CUANTOS DADOS?1
1
CUANTOS DADOS?1,5
```

```
CUANTOS DADOS?0
CUANTOS DADOS?10
3,4,3,6,3,2,1,6,4,3
CUANTOS DADOS?2
6,6
CUANTOS DATOS?3
2,5,4
CUANTOS DADOS?7
3,5,6,5,4,2,4
CUANTOS DADOS?5
6,3,6,2,5
CUANTOS DADOS?6
5,4,2,3,3,3
```

Discusión de resultados

Los resultados serán objeto de visualización. Cuando pedimos 0 ó 1,5 dados, puesto que ambos números no son admisibles, reaparecerá el mensaje CUANTOS DADOS.

OBSERVACIONES PARA LA PROGRAMACION

La comprobación de errores de la introducción se realiza en la línea 70. El programa es sencillo y corto. La línea 90 indica cómo transformar el número aleatorio generado por RND desde el margen 0-1 a la gama de enteros entre 1 y 6.

LISTADO DEL PROGRAMA

```
 10 CLS
 20 PRINT "DADOS MULTIPLES"
 30 PRINT
 40 PRINT "CUANTOS DADOS?";
 50 INPUT A
 60 PRINT A
 70 IF A<>INT A OR A<1 THEN GOTO 40
 80 FOR I=1 TO A
 90 PRINT INT (6*RND)+1;
100 IF I=A THEN GOTO 120
110 PRINT ",";
120 NEXT I
130 GOTO 30
```

25. Cuenta corriente bancaria de dos usuarios

DESCRIPCION DEL PROGRAMA

Se trata de un programa para el control financiero de una familia que tiene una cuenta corriente con dos titulares. Actualmente suele darse el caso de que trabajen los dos esposos y se plantea, con frecuencia, el problema de cómo controlar una cuenta corriente conjunta. Ambas partes pueden disponer de talonarios individuales en los que registran por separado todas sus transacciones tales como depósitos, cheques extendidos contra la corriente y transferencias entre los dos esposos. Cuando el banco envía un estado de cuenta a finales de mes, el siguiente programa ayudará a saldar los dos talonarios. Las ecuaciones utilizadas son las siguientes:

Saldo esposo n.º 2 = saldo mes anterior del esposo n.º 2 + depósitos del esposo n.º 2 – cheques compensados del esposo n.º 2 + transferencias del esposo n.º 1 al n.º 2, también

Saldo esposo n.º 2 = saldo bancario corriente – saldo talonario n.º 1 – cheques no compensados n.º 1 – interés devengado (para cuentas AHORA) + gastos bancarios.

El saldo del esposo n.º 2 se calcula por el programa de dos formas y se comparan los resultados. Si son los mismos, la cuenta estará saldada. Si son diferentes, habrá un error en el estado bancario, o más probablemente, en uno de los asientos del talonario.

INSTRUCCIONES

Al pulsar al tecla RUN se emite una llamada orientativa correspondiente a las órdenes siguientes:

Z — Borra todos los registros, comienza programa
X — Correspondiente al saldo anterior del esposo n.º 2
X — Correspondiente a cada depósito para la cuenta conjunta por el esposo n.º 2
V — Correspondiente a cada cheque compensado por el esposo n.º 2
B — Correspondiente a cada transferencia interna desde la cuenta del esposo 1 a la del n.º 2; utilizar valores negativos para las transferencias desde el n.º 2 al n.º 1

N — Correspondiente al saldo del talonario corriente del esposo n.º 1
M — Correspondiente a cheques no compensados extendidos por el esposo n.º 1
A — Correspondiente al saldo bancario corriente
S — Correspondiente al interés devengado (cuentas corrientes bancarias de AHORA)
K — Correspondiente a los gastos bancarios, si los hubiera
D — Visualiza el saldo del esposo n.º 2 calculado en ambas formas.

EJEMPLO

Enunciado del problema

El estado de la cuenta bancaria mensual acaba de llegar y muestra un saldo en la cuenta conjunta de $1234.45. El esposo n.º 1 tiene un saldo del talonario de $751.23 y dos cheques extendidos por el n.º 1 por las cantidades $15.25 y $10.00 no estaban todavía compensados. El banco abonó en la cuenta unos intereses de $6.75. El esposo n.º 2 extendió, durante ese mes, tres cheques por las cuantías de $250.00, $125.00 y $50.00, pero el último cheque no estaba todavía compensado. El esposo n.º 2 tenía depósitos abonados en cuenta de $150.00 y $175.00 y una transferencia interna del esposo n.º 1 por la cuantía de $300.00. El saldo del talonario anterior del esposo n.º 2 era de $203.22 y los gastos bancarios llegaron a ser exactamente de $2.00. Calcular el saldo del talonario corriente del esposo n.º 2 y ver si la cuenta está saldada.

Ejemplo de ejecución

```
CUENTA CORRIENTE DE 2 PERSONAS

INTRO.Z,X,C,V,B,N,M,A,S,K O D?X
SALDO ANTERIOR NO.2?203.22

INTROD.Z,X,C,V,B,N,M,A,S,K O D?C
DEPOSITOS NO. 2?150,TOT=150

INTROD.Z,X,C,V,B,N,M,A,S,K O D?C
DEPOSITOS NO. 2?175,TOT=325

INTROD.Z,X,C,V,B,N,M,A,S,K O D?V
CHEQUES COMPENSADOS NO. 2?250,TOT=250

INTRO.Z,X,C,V,B,N,M,A,S,K O D?V
CHEQUES COMPENSADOS NO. 2?125,TOT=375
```

```
INTROD.Z,X,C,V,B,N,M,A,S,K O D?B
TRANSFERENCIAS 1 A 2?300,TOT=300

INTROD.Z,X,C,V,B,N,M,A,S,K O D?N
SALDO NO. 1?751.23

INTROD.Z,X,C,V,B,N,M,A,S,K O D?M
CHEQUES PENDIENTES NO. 1?15.25,TOT=15.25

INTROD.Z,X,C,V,B,N,M,A,S,K O D?M
CHEQUES PENDIENTES NO. 1?10,TOT=25.25

INTROD.Z,X,C,V,B,N,M,A,S,K O D?A
SALDO BANCARIO?1234.45

INTROD.Z,X,C,V,B,N,M,A,S,K O D?S
INTERES?6.75

INTROD.Z,X,C,V,B,N,M,A,S,K O D?K
CARGAS BANCARIAS?2,TOT=2

INTROD.Z,X,C,V,B,N,M,A,S,K O D?D
SALDO NO. 2=453.22,453.22
CUENTA SALDADA
```

Discusión de los resultados

La cuenta está afortunadamente saldada; ambos cálculos llevan al mismo resultado. Obsérvese que el cheque de $50 extendido por el esposo n.º 2, que no estaba todavía compensado, no se tuvo que introducir.

OBSERVACIONES SOBRE LA PROGRAMACION

La bifurcación se realiza a través de la función INKEY$ y luego, la entrada se compara con las órdenes reconocidas Z, X, C... La línea 580 se refiere al error de redondeo del Timex/Sinclair 1000. Obsérvese que en la subrutina, que comienza en la línea 1000, todas las variables están puestas a 0. ¿Por qué no utilizan la orden CLEAR en su lugar? La razón es que las sentencias tales como las que están en la línea 190 o 250 proporcionarían el error 2 (variable no definida).

LISTADO DEL PROGRAMA

```
 10 CLS
 15 PRINT "CUENTA CORRIENTE DE 2 PERSONAS"
 20 SLOW
 30 GOSUB 1000
 40 PRINT
 45 PRINT "INTROD.Z,X,C,V,B,N,M,A,S,K,O,D?";
 50 IF INKEY$="" THEN GOTO 50
 55 PAUSE 5
 60 LET A$=INKEY$
 65 IF A$=INKEY$ THEN GOTO 65
 70 PRINT A$
 90 IF A$<>"Z" THEN GOTO 110
 95 GOSUB 1000
100 PRINT "SISTEMA COMPENSADO"
105 GOTO 40
110 IF A$<>"X" THEN GOTO 160
120 PRINT "SALDO ANTERIOR NO. 2?";
130 INPUT X
140 PRINT X
150 GOTO 40
160 IF A$<>"C" THEN GOTO 220
170 PRINT "DEPOSITOS NO. 2?";
180 INPUT P
190 LET C=C+P
200 PRINT P;",TOT=";C
210 GOTO 40
220 IF A$<>"V" THEN GOTO 280
230 PRINT "CHEQUES COMPENSADOS NO. 2?";
240 INPUT Y
250 LET V=V+Y
260 PRINT Y;",TOT=";V
270 GOTO 40
280 IF A$<>"B" THEN GOTO 340
290 PRINT "TRANSFIERE 1 A 2?";
300 INPUT L
310 LET B=B+L
320 PRINT L;",TOT=";B
330 GOTO 40
340 IF A$<>"N" THEN GOTO 390
350 PRINT "SALDO NO.1?";
360 INPUT N
370 PRINT N
380 GOTO 40
```

```
 390 IF A$<>"M" THEN GOTO 450
 400 PRINT "CHEQUES PENDIENTES NO.1?";
 410 INPUT J
 420 LET M=M+J
 430 PRINT J;",TOT=";M
 440 GOTO 40
 450 IF A$<>"A" THEN GOTO 490
 455 PRINT "SALDO BANCARIO?";
 460 INPUT A
 470 PRINT A
 480 GOTO 40
 490 IF A$<>"S" THEN GOTO 540
 500 PRINT "INTERES?";
 510 INPUT S
 520 PRINT S
 530 GOTO 40
 540 IF A$<>"D" THEN GOTO 610
 550 LET F=X+C-V+B
 560 LET G=A-N-M-S+K
 570 PRINT "SALDO NO. 2=";F;",";G
 580 IF ABS (F-G)<.001 THEN PRINT
     "CUENTA SALDADA"
 590 GOTO 40
 610 IF A$<>"K" THEN GOTO 660
 620 PRINT "CARGAS BANCARIAS?";
 630 INPUT F
 640 LET K=K+F
 650 PRINT F;",TOT=";K
 655 GOTO 40
 660 PRINT "REPEAT"
 670 GOTO 40
1000 LET C=0
1010 LET V=0
1020 LET B=0
1030 LET M=0
1040 LET K=0
1050 LET X=0
1060 LET A=0
1070 LET N=0
1080 LET S=0
1090 RETURN
```

APENDICE

Técnicas y artificios de programación

Al escribir y depurar los programas de este libro utilicé varias técnicas y artificios de programación que desearía compartir con otros usarios del Timex/Sinclair 1000 para evitarles algún desconsuelo y frustración. Se citan a continuación:

1. Redondeo

Si trabaja con números de coma flotante no tiene una decisión sobre igualdades de dos números (v. g., IFX = YTHEN...). El TS 1000 trata el redondeo de forma extraña; v.g.

```
100 LET X = 3.25 – 3 –.25
110 PRINT X
```

visualizará 1.164... E – 10. Por consiguiente, ha de tomarse siempre el valor absoluto de la diferencia y compararle con un número pequeño.

2. Funcionamiento fuera de memoria

Cuando escriba un programa largo que haga que el Timex/Sinclair 1000 funcione fuera de memoria, la computadora comenzará a «agonizar» y le hará la vida imposible. Uno de los primeros síntomas de la «escasez» de memoria es cuando la computadora ignora la función EDIT (Desplazado, «SHIFT», 1). Sin embargo, cuando el Timex/Sinclair 1000 comparte la visualización y la memoria de programa puede resolver este problema. Por ejemplo, si quiere efectuar la edición correctora de la línea 350 mientras está trabajando fuera de memoria, haga lo siguiente:

LIST 350	El listado aparece en la pantalla con la línea 350 en la parte superior
CLS	La pantalla se pone en blanco
EDIT	Inmediatamente después, la línea 350 aparece en la parte inferior de la pantalla preparada para su edición correctora

3. Bifurcaciones múltiples

El 1000 carece de una GOTO calculada. Para subsanar esta deficiencia, numere las diversas bifurcaciones de su pograma en múltiplos de 100 y utilice la sentencia

```
GOTO 100* A
```

La variable A debe ser, por supuesto, un entero.

4. Salida de un lazo

Si se queda «atascado» durante la ejecución en un lazo «apretado» con una sentencia INPUT en el mismo, quizás no pueda salir del lazo sin tener que recurrir a desenchufar la computadora. BREAK se interpretará por INPUT como un espacio. Si la entrada prevista es numérica, todavía puede salirse del lazo tecleando, por ejemplo, Shift Enter (función) LN 0 y obteniendo un error «A». Si la entrada prevista es alfanumérica, su única esperanza es probar a llenar la pantalla y obtener un error «5».

5. Sentencia INPUT

Preceder a todas las sentencias INPUT con una llamada orientativa. Una sentencia INPUT no produce un signo de interrogación, de forma automática, como en algunas otras versiones del BASIC. Siga una sentencia INPUT con una PRINT para visualizar lo que acabe de introducir. De no ser así, nunca lo sabrá. Una secuencia típica podría ser la siguiente:

```
10 PRINT «INTRODUCIR X?»;
20 INPUT X
30 PRINT X
```

6. Utilización de INKEY$

INKEY$ se utiliza como un medio adecuado para la bifurcación del programa. Dentro de un lazo con INKEY$, la computadora debe estar en el modo SLOW (o lento) o el programa debe tener sentencias PAUSE; de no ser así, la pantalla se mantendrá en blanco sin ningún mensaje o llamada orientativa que sea visible. En este caso, hay una secuencia «a prueba de uso indebido» que ha de detectarse; por ejemplo, la clave «X». Trabajará en los dos modos SLOW y FAST (lento y rápido):

```
100 PRINT «CUAL ES EL SIGUIENTE?»;
110 PAUSE 10
120 IF INKEY$ = “ ” THEN GOTO 110
130 LET A$ = INKEY$
140 PRINT A$
150 PAUSE 10
160 IF A$ = INKEY$ THEN GOTO 150
170 IF A$ < > “X” THEN GO TO 100
180 REM “X” SE HA DETECTADO
```

Programa n.º 8. Cálculos del desplazamiento relativo para la programación en código máquina.

Introducir los dígitos hexadecimales A a F en mayúsculas o sustituir las líneas 280, 290 y 370 y añadir las líneas 281, 291 y 371 como sigue:

```
280 IF g < =9 THEN LET e$ = CHR$(g + 48)
281 IF g > 9 THEN LET e$ = CHR + (g + 55)
290 IF g < 9 THEN LET w$ = CHR$(z – 16 * g + 48)
291 IF g > 9 THEN LET w$ = CHR$(z – 16 * g + 55)
370 IF CODE c$(1-i) < = 57 THEN LET r = CODE c$(1-i) – 48
371 IF CODE c$(1-i) > 57 THEN LET r = CODE c$(1-i) – 55
```

Referencia y lectura suplementaria

1. *Science and Engineering Sourcebook,* Cass R. Lewart, Prentice-Hall/MicroText. Colección de programas para las computadoras personales TRS-80 PC/Shap PC-1211.

2. *Handbook of Mathematical Functions*, U.S. Department of Commerce, National Bureau of Standards, editado por Milton Abramowitz e Irene A. Stegun. La mejor colección de fórmulas matemáticas.

3. *Introduction to Scientific Computing*, Gabriel A. Pall, Appleton-Century Crofts. Una buena explicación del método Runge-Kutta y otras aproximaciones numéricas.

4. *Reference Data for Radio Engineers,* International Telephone and Telegraph Corporation, 4.ª y 5.ª ediciones. Excelente compendio de fórmulas relativas a la ingeniería electrónica.

5. «Predicting Queue Performance on a Programmable Handheld Calculator», Ronald Zussman, *Computer Design.* Agosto 1978. Un programa de parámetros de colas para TI-59.

6. *Fundamentals of Queuing Theory,* Donald Gross y Cark M. Harris, J. Wiley & Sons. Una descripción profunda y derivación de ecuaciones de colas.

7. «Forecast Computer System Reliability with a Handheld Programmable Calculator», Ronald Zussman, *Computer Design*, marzo 1979. Un programa de parámetros de fiabilidad para TI-59.

8. «Reliability Computations on a Handheld Programmable Calculator», Cass R. Lewart, *Computer Design*, noviembre 1979. Un programa de parámetros de fiabilidad para HP-67.

9. «Calc Program Assembles Machines Code», Cass R. Lewart y Daniel S. Lewart, *Machine Design*, 12 de marzo de 1981. Programa de cálculos de la desviación relativa para HP-67.

10. «Pocket Computer Solves for LC Resonance Using BASIC», Cass R. Lewart, *Electronics,* 16 de junio de 1981, p. 189.

11. «Pocket Computer Tackles Queuing Problems in BASIC», Cass R. Lewart, *Electronics,* 22 de septiembre de 1981, p. 157.

12. «Pocket Computer Scales Computer Generat de Plots», Cass R. Lewart, *Electronics*, 15 de diciembre de 1981, p. 159.

Indice

Adaptación impedancia, 12
Ajuste curvas, 316
Apantallamiento, 12
Artificios de programación, 110
Atenuación, 7
Atenuador adaptación, 7
Atenuador pérdidas mínimas, 7

Base binaria, 22

Capacidad, 1
Coeficientes binómicos, 45
Coeficientes de Fourier, 16
Combinaciones, 45
Comillas dentro de comillas, 112
Condiciones iniciales, 67
Constante dialéctrica, 12
Conversión base, 22, 26
Conversión hexadecimal, 22, 26
Criba de Eratosthenes, 29
Criptografía, 29
Cuenta corriente bancaria, 105
Curva francesa, 36

Desplazamiento relativo, 32
Desviación estándar, 49
Desviantes pseudo aleatorias, 57
Dimensionamiento cadenas, 112
Direccionamiento relativo, 32
Disponibilidad, 90
Distribución binómica, 52
Distribución exponencial, 83
Distribución gaussiana, 49, 57
Distribución hipergeométrica, 52
Distribución media, 57
Distribución normal, 49, 57
Distribución uniforme, 57

Ecuaciones de las colas, 83
Ecuaciones diferenciales, 67
Ecuaciones trascendentes, 71
Enlace de datos, 18
Ensamblador, código, 32
Escala afinada, templada, 96
Escalas legibles, 41
«Escasez» de memoria, 110
Extrapolación, 36

Factores primos, 29
Factoriales, 45
Fiabilidad, 89
Fracciones binarias, 80
Fracciones decimales, 80
Función de alisado, 36
Función de error, 18, 49
Función impar, 16
Función par, 16
Función periódica, 62

GOTO calculada, 111

Impedancia característica, 12
Inductancia bobina, 1
Inicialización, 112
Inkey$, 111
Inteligencia artificial, 99
Iteración de Newton, 71
Interpolación, 36

Juego de cara o cruz, 99
Juego de estrategias, 99
Juegos de posibilidades, 103

Lenguaje máquina, 32
Línea de transmisión, 12

Microprocesador, 32
Modulación por desplazamiento de amplitud, 18
Modulación por desplazamiento de fase, 19
Modulación por desplazamiento de frecuencia, 18
Modulación por desplazamiento de pulsos, 18
Modulación diferencial, desplazamiento de fase, 19
Modulación OOK, 18
Muestreo, universo, 52

Numeración polaca inversa (RPN), 75
Número sentencia, edición, 112
Números complejos, 75

Octavas, 96

Parámetros bobina, 1
Permutaciones, 45
Pila (RPN), 75
Polinomios, 36
Polinomios de Lagrange, 36

Redondeo, 110
Redundancia, 89
Referencias, 113
Registros aritméticos, 75
Runge-Kutta, 67

Saldo bancario, 105
Señal/ruido, 18
Simulación de Monte-Carlo, 57
Sirvientes, 83
Supresión memoria intermedia, 112

Tasa de errores de bits, 18
Tasa de fallos, 89
Tasa de llegada, 83
Tasa servicio, 83
Teoría de los números, 22
Tiempo de espera, 83
Tiempo medio antes de fallos (MTBF), 89
Tiempo medio hasta reparación (MTTR), 89
Tiempo de respuesta, 83
Tirada de dados, 103
Tonos musicales, 96
Transmisión de datos, 18
Trazado gráficos, 36, 41

Utilización, 84
Utilización instalaciones, 84

YAH-T-ZEE, 103

OTRAS OBRAS DE INTERES PUBLICADAS POR OSBORNE/McGRAW-HILL

ADAM OSBORNE: Guía del comprador de sistemas de gestión
ADAM OSBORNE: Guía del ordenador personal PET/CBM
ANNIE FOX: BASIC básico. Guía para principiantes
CASTLEWITZ: Introducción al Visicalc
ETTLIN: Introducción al Wordstar (2.ª ed.)
JOHN HEILBORN: Programas para ciencias e ingeniería. Edición APPLE II
JOHN HEILBORN: VIC 20. Guía del usuario
LON POOLE: Algunos programas de uso común en BASIC
LON POOLE: Algunos programas de uso común en PASCAL
LON POOLE: Algunos programas de uso común en BASIC. Edición APPLE II
LON POOLE: Algunos programas de uso común en BASIC. Edición ATARI
LON POOLE: Algunos programas de uso común en BASIC. Edición IBM/PC
LON POOLE: Algunos programas de uso común en BASIC. Edición PET/CBM
LON POOLE: Algunos programas de uso común en BASIC. Edición TRS-80
LON POOLE: APPLE II. Guía del usuario
LON POOLE: Programas prácticos en BASIC
LON POOLE: Programas prácticos en BASIC. Edición APPLE II
LON POOLE: Programas prácticos en BASIC. Edición IBM/PC
LON POOLE: Programas prácticos en BASIC. Edición TRS-80
LON POOLE: Programas prácticos en PASCAL
LYLE J. GRAHAM: IBM/PC. Guía del usuario
ROBERT MOTTOLA: Programación en lenguaje ensamblador para el APPLE II
TOM HOGAN: Sistema operativo CP/M. Guía del usuario (2.ª ed.)

DISCOGUIAS PUBLICADOS POR OSBORNE/McGRAW-HILL

CURTIS A. INGRAHAM: Discoguía para CP/M
DAVID A. WILSON: Discoguía para IBM/PC
DAVID A. WILSON: Discoguía para VISICALC
JOHN TAYLOR: Discoguía para ATARI 400/800
ZELDA GIFFORD: Discoguía para APPLE II

OTRAS OBRAS DE INTERÉS PUBLICADAS POR BYTE-BOOKS McGRAW-HILL

CLARCIA: Construya una computadora basado en el Z-80 (Guía de diseño y funcionamiento)
KAMINS: Usted y la "Microcomputadora" (Una introducción humanizada a la microinformática)
MORGAN: Introducción al microprocesador 8086/8088 (16 bit)

ISBN: 968-451-603-7